UNIVERSIDAD NACIONAL DE ASUNCIÓN

Facultad Politécnica

"Diseño e implementación de una solución integrada de recolección y análisis predictivo de datos open source utilizando dispositivos móviles inteligentes"

Trabajo Final de Grado presentado por

Luis Rodríguez
y Marcio Duarte

como requisito
para obtener el título de Ingeniero
en Informática

Orientadores:

D.Sc. Christian Schaerer
MSc. Santiago Gómez
Dra. Antonieta Rojas de Arias

San Lorenzo - Paraguay. Diciembre de 2017

Este trabajo está dedicado a nuestras familias,
por el apoyo incondicional brindado
durante nuestra formación académica
y por ser ejemplos de esfuerzo y dedicación.

Agradecimientos

A los profesores Christian Schaerer y Santiago Gómez por la orientación y consejos que nos ayudaron a llevar adelante este trabajo.

A la Dra. Antonieta Rojas de Arias por brindarnos la oportunidad de ser partícipes de colaborar con su labor investigativa.

Al Laboratorio de Computación Científica y Aplicada por el espacio brindado para realizar los estudios.

A nuestros profesores y compañeros de la FP-UNA que nos acompañaron durante nuestro proceso de formación.

A nuestros familiares por el apoyo incondicional durante esta etapa de nuestras vidas.

Resumen

En este trabajo proponemos una solución de recolección y análisis de datos que parte de la utilización de dispositivos móviles como herramientas de relevamiento de la información utilizando formularios electrónicos y se completa con el posterior análisis en el cual se apunta a la extracción de conocimiento y creación de modelos de predicción a partir de los datos relevados de las encuestas.

Realizamos una comparación de herramientas de recolección de datos open source disponibles, seleccionando la suite Open Data Kit (ODK) debido a sus capacidades multimedia y su utilización exitosa en otros proyectos de recolección de datos.

Se hace uso de las soluciones propuestas en el caso de estudio de recolección de datos sobre infestación intra y peridomiciliar de la enfermedad de Chagas que fue realizada en localidades indígenas de los departamentos de Boquerón y Presidente Hayes en el Chaco paraguayo.

Aplicamos técnicas de clasificación y modelos de predicción sobre los datos recogidos utilizando una API creada para el efecto, que utiliza Weka como librería de aprendizaje de máquina, dando como resultado un clasificador para predecir la presencia de insectos vectores en hogares, sirviendo además como herramienta para determinar factores que favorecen la infestación en viviendas. Además, comparamos los resultados con estudios realizados sobre estos factores, obteniendo resultados similares.

La suite de herramientas ofrecidas por ODK ofrecen una solución efectiva y escalable al problema de la recolección de datos con posibilidades de ser aplicada en distintos ámbitos con énfasis en salud pública. El análisis predictivo de datos añade un valor agregado al convertirse en una solución valiosa para investigadores, útil para la toma de decisiones.

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Lista de acrónimos

API Application Program Inteface

CEDIC Centro para el Desarrollo de la Investigación Científica

CPU Central Processing Unit

CSV Comma Separated Values

DGEEC Dirección General de Estadísticas, Encuestas y Censos

DGVS Dirección General de Vigilancia de la Salud

EVD Ebola Virus Disease

GLONASS Globalñaya Navigatsionnaya Sputnikovaya Sistema

GPS Sistema de Posicionamiento Global

GSM Global System for Mobile communications

HSPA High-Speed Packet Access

HTML HyperText Markup Language

IVR Interactive Voice Response

JSON JavaScript Object Notation

KML Keyhole Markup Language

LTE Long Term Evolution

MSPyBS Ministerio de Salud Pública y Bienestar Social

ODK Open Data Kit

PC Personal Computer

PNCECh Programa Nacional de Control de la Enfermedad de Chagas

UNA Universidad Nacional de Asunción

UW Universidad de Washington

$W3C$ World Wide Web Consortium

WAR Web Application Archive

WFP Wold Food Programme

$WiFi$ Wireless Fidelity

XML eXtensible Markup Language

Capítulo 1

Introducción

La recolección de datos es fundamental para la investigación, y a menudo es un factor prominente a la hora de evaluar el costo y el éxito del mismo.

Se tiene evidencia de que los organismos del estado encargados de recoger datos de salud pública y variables socioeconómicas [Rec14; Col12; Pob12] realizan en su mayoría la recolección de datos mediante técnicas tradicionales de papel y lápiz a pesar de sus ineficiencias, como son el almacenamiento de los formularios, errores de transcripción y la disponibilidad de datos. Las instituciones que utilizan software de recolección de datos lo llevan a cabo mediante herramientas propietarias o soluciones a medida, lo que representa un esfuerzo económico importante en licencias y desarrollo de software. [Cona].

Actualmente, existe una tendencia de automatización de procesos de recolección de datos en organizaciones [Koe], con la finalidad de acelerar los mismos y maximizar la confiabilidad de los datos. A este proceso de recolección, además se le suma el posterior análisis de información, el cual a través de la aplicación de técnicas de minería de datos y aprendizaje de máquinas, tiene el potencial de producir información valiosa y novedosa para investigadores, permitiéndoles predecir variables y tener una visión ampliada de las mismas, y de esta manera, ayudándolos a tomar mejores decisiones.

Las técnicas de aprendizaje de máquina permiten la creación de modelos de predicción, en los cuales se pueden visualizar la interacción de las variables recogidas utilizando algoritmos de clasificación, destacando aquellas variables relevantes o con relación directa al fenómeno a predecir de un caso de estudio.

Si bien la modalidad de recolección de datos móviles se puede aplicar a cualquier disciplina, el presente trabajo toma como caso de estudio un problema relacionado a la salud pública, debido a que es un área de especial interés de los investigadores, y en donde a menudo se están realizando trabajos de importancia para la población en general utilizando métodos tradicionales de recolección.

El presente trabajo plantea: la integración y desarrollo de herramientas open source para la recolección de datos y el posterior análisis de los mismos aplicando técnicas de minería de datos y aprendizaje de máquina.

1.1. Antecedentes

En Paraguay las instituciones públicas y empresas privadas realizan encuestas principalmente utilizando papel y lápiz, si bien se tiene conocimiento de la utilización de PDAs por parte de la Dirección General de Estadísticas, Encuestas y Censos (DGEEC) utilizados para el censo poblacional realizado en el año 2012 [Col12; Pob12], y para la Primera Encuesta Nacional de Factores de Riesgo de Enfermedades No Transmisibles llevado a cabo en el año 2011 por la Dirección de Vigilancia de Enfermedades No transmisibles dependiente del Ministerio de Salud Pública y Bienestar Social (MSPyBS) [Sal12] no se tiene evidencia de una normativa que impulse su utilización, ni tampoco de una base documental que permita a las organizaciones el despliegue de dichas metodologías.

Además, existen instituciones que realizan licitaciones públicas con el objeto de encontrar una solución al problema de recolección de datos, lo que se traduce en gastos de licencia de software propietario ofrecido por empresas privadas, algunas de ellas hechas a medida para la institución solicitante [Cona; Conb].

Como caso particular de estudio, el Centro para el Desarrollo de la Investigación Científica (CEDIC) con la colaboración del Programa Nacional de Control de la Enfermedad de Chagas (PNCECh), del MSPyBS tienen como fin prevenir la mortalidad y disminuir las pérdidas socio-económicas debidas a la enfermedad de Chagas [SEN; Des11]. Con el objeto de recoger variables socio-económicas, comportamientos, actitudes y creencias sobre la enfermedad de Chagas, se realizan encuestas a los hogares en las regiones afectadas [Arr+14; Sal11].

La encuesta se realiza utilizando un formulario en papel. Los encuestadores completan los datos correspondientes a los hogares y registran sus coordenadas utilizando un dispositivo para conocer su posición en el Sistema de Posicionamiento Global (GPS).

Una vez que todos los hogares hayan sido encuestados, los formularios en papel regresan hasta la CEDIC, en donde los mismos son digitalizados en planillas electrónicas para su posterior análisis.

1.2. Planteamiento del problema

Desde inicios del Siglo 20, las tareas de toma de datos estadísticos a través de censos y encuestas se han estado realizando sobre papel en forma de cuestionarios. El papel como medio de soporte de los datos obliga a usar diversas técnicas para procesar los datos, habiéndose empleado planillas de sumarización, máquinas perforadoras y tabuladoras, máquinas de calcular manuales y eléctricas, hasta llegar a las computadoras. Hoy se dispone de nuevas tecnologías pero en gran medida todavía ligadas al papel. El estado actual de las tareas de recolección y procesado de datos de encuestas presenta los siguientes problemas:

Largas esperas para tener acceso a la información

Para acceder a los datos recogidos, los formularios en papel deben llegar al centro de procesamiento para posteriormente ser utilizados, muchas veces los mismos deben

Proyecto
Origen de las reinfestaciones intra y peridomiciliarias de *Triatoma infestans* en viviendas indígenas del Chaco Paraguayo

Cuestionario Domiciliar

Localidad: Campo Largo - 6 de Octubre Grupo Nro.:_____

Nombre del jefe de Familia: Gerardo Oviedo

Fecha: 290508 Número de la vivienda: 86 GPS: 22.53.17 3086 060.14.04

1. Datos de la Vivienda

Nombre del encuestado: _____

Número de cuartos en la vivienda: 1

Número de personas que viven en la vivienda 3

Antigüedad de la vivienda: 5 (años).

2. Tipo de estructura

Pared	**Techo**	**Piso**
☐ Ladrillo	☐ Paja	☒ Tierra
☐ Barro	☐ Tejas	☐ Cemento
☐ Tronco	☒ Zinc	☐ Ladrillo
☐ Pared francesa	☐ Tronco	☐ Cerámica
☐ Ladrillo revocado	☐ Otro_____	☐ Otro_____
☒ Barro revocado		
☐ Pared francesa revocada		

3. Prevalencia de triatominos en las Viviendas

3.1 Domicilio:

☒ **Captura negativa** ☐ **Captura positiva:**
- ☐ Adultos
- ☐ Ninfas
- ☐ Huevos embrionados
- ☐ Huevos eclosionados
- ☐ Otros:_____

3.2 Lugar de captura (especificar tipo de estructura)

- ☐ Paredes
- ☐ Piso
- ☐ Techo
- ☐ Camas
- ☐ Ropas
- ☐ Otros_____

3.3 Peridomicilio:

☐ **Captura negativa** ☒ **Captura positiva:**
- ☐ Adultos
- ☒ Ninfas
- ☐ Huevos embrionados
- ☐ Huevos eclosionados
- ☐ Otros _____

3.4 Lugar de captura (especificar el sitio exacto dentro del lugar):

- ☒ Gallinero
- ☐ Deposito
- ☐ Galpón
- ☐ Corral
- ☐ Materiales acumulados
- ☐Otros_____

Observaciones: _____

Figura 1.1: Formulario utilizado para la encuesta a los hogares

recorrer grandes distancias, o lugares poco accesibles.

Errores en la entrada de datos

El hecho de no existir una validación al momento del llenado del formulario se traduce en una fuente de posibles errores en la carga de los datos.

Potencial pérdida de datos

Dada la naturaleza del papel, se corre el riesgo de la pérdida total o parcial de los datos recolectados.

Costo de procesamiento de los datos recogidos

Una vez que los formularios en papel llegan al centro de datos este debe ser digitalizado manualmente, durante dicho proceso se pueden producir errores de transcripción, lo que también añade un punto de fallo. Existen casos en donde los establecimientos no cuentan con infraestructura adecuada para la digitalización de los mismos, lo cual conlleva a que el análisis de los datos se realice de forma manual. Si la recolección de datos implica grandes volúmenes de datos el procesamiento manual resulta impracticable.

Costos de distribución altos

Para poner en marcha la recolección de datos los formularios deben ser impresos (sin mencionar el impacto ambiental de los mismos [Mag]) y distribuidos a las regiones en donde se realizará la toma de información, lo cual de nuevo implica el recorrido de grandes distancias y largas esperas para que la recolección de datos pueda llevarse a cabo.

Estructura de datos heterogénea

Cada encuesta cuenta con una estructura de base de datos en particular, este hecho hace que no se pueda acceder a los datos de una manera unificada, lo que dificulta la posibilidad de realizar análisis estadísticos o minerías de datos. La dificultad reside en que se debe conocer la estructura de los datos subyacente para cada formulario en particular,

Costo de licencias de software

Las instituciones que implementaron el uso de formularios digitales para la recolección de datos incurren en costos de licencias de software propietarias que son adquiridas desde empresas privadas mediante licitaciones.

1.3. Objetivos

1.3.1. Objetivo general

- Diseño e implementación de una solución integrada de recolección y análisis predictivo de datos open source utilizando dispositivos móviles inteligentes.

1.3.2. Objetivos específicos

- Evaluar herramientas open source existentes para la recolección de datos móviles.
- Probar la confiabilidad la herramienta seleccionada realizando encuestas en campo.
- Documentar el proceso requerido para el despliegue y utilización de la solución propuesta.
- Construir herramientas para la extracción de conocimiento a partir de los datos recogidos.
- Utilizar técnicas de aprendizaje de máquina para la creación de modelos de clasificación como herramientas predictivas.
- Integrar las herramientas utilizadas de forma a crear una plataforma de recolección y análisis predictivo de datos.

1.4. Importancia

La recolección de datos es un proceso requerido en distintos casos de estudio para ofrecer una perspectiva de la situación actual a los investigadores. Utilizando métodos tradicionales, a mayor cantidad de datos se invierte mayor tiempo en procesarlos; por otro lado, automatizando este proceso, el procesamiento de los mismos se realiza cuasi inmediatamente, permitiendo la disponibilidad para el análisis en menor tiempo. Con esto, los investigadores podrán centrar su esfuerzo en el análisis de datos más que en la recolección, permitiéndoles tener un campo de visión más amplio de los datos manejados, debido a las capacidades multimedia de los dispositivos a utilizar, obteniendo mayor precisión a la hora de tomar decisiones basadas en estos análisis. La herramienta de predicción de datos puede ser fundamental para visibilizar los casos de estudio en donde se requieran tomar acciones en base a un resultado específico, ahorrando de esta manera tiempo y recursos. Por ejemplo, anticipar brotes de una enfermedad o determinar personas vulnerables a un fenómeno climático. Con la predicción, además, se pueden identificar los factores que inciden para que se produzca un fenómeno en particular, ayudando de esta manera a un mejor entendimiento del problema.

1.5. Justificación

Las características de conectividad de los teléfonos inteligentes, tales como acceso a internet (datos móviles y WiFi), sensores GPS y captura de multimedia (fotografías, sonido, vídeo) los hace ideales para su uso en la captura de datos [Med+15].

Reducir la cantidad de tiempo entre la toma de datos y el análisis de los mismos permitirá la toma de decisiones más rápidas, y con mayor precisión [Nju+14; Zha+12].

Dicha precisión radica especialmente en la validación de datos in-situ, evitando además problemas de grafías no legibles como los que se evidencia en la planilla de semanal de notificación obligatoria de la Dirección General de Vigilancia de la Salud (DGVS), dependiente del MSPyBS en donde solo el 75 % de los formularios se rellena de manera correcta [Rec14].

Al utilizar herramientas open source, ayudarán a reducir costos de adquisición de software propietario en instituciones que precisen soluciones de esta naturaleza.

Una vez finalizada la etapa de captura de datos, se podrá disponer de herramientas que faciliten la visualización y análisis de la información recogida, utilizando gráficos, mapas, tablas y la información multimedia que se ha recogido. Además, con las herramientas de predicción y extracción de conocimiento se podrán obtener conclusiones que ayudarán a entender los diversos aspectos de un problema en particular.

El uso de esta metodología beneficiará principalmente a los investigadores e instituciones agilizando los procesos tradicionales, contando con herramientas que le servirán de respaldo para la toma de decisiones que podrían ser aplicadas, por ejemplo, a políticas públicas que busquen el bienestar general de la población.

1.6. Originalidad

Actualmente se disponen de varias plataformas de recolección de datos tanto para computadores de escritorio como para dispositivos móviles, en su mayoría productos de licencia paga. Si bien varias instituciones ya utilizan una herramienta de recolección móvil, éstas soluciones están diseñadas para resolver un problema en particular (lo que se conoce comúnmente como solución a medida), o están incurriendo en costos por licencias de utilización de herramientas.

Hemos realizado una investigación de la situación del mercado de software de recolección de datos con las siguientes características: que esté disponible para plataformas móviles y con licencia Open Source. Esto con el fin de encontrar una herramienta versátil, modificable a medida y sin invertir en gastos por licencias de software. Se pretende que con éste trabajo sentar las bases para una plataforma común, genérica, y multipropósito que pueda ser fácilmente accedida y desplegada para instituciones o personas que lo requieran.

En capítulos posteriores describimos nuestra selección de estas herramientas, que cumplen con la tarea de facilitar la recolección de datos, explorando además herramientas de análisis de datos que puedan enriquecer la captura de datos. Nuestra propuesta consiste en utilizar una herramienta Open Source para la recolección de

datos y preparar los mismos para luego utilizar técnicas de aprendizaje de máquina, generando modelos predictivos de modo a que los datos recogidos produzcan información útil para la toma de decisiones de investigadores, de esta forma, consiste en una solución completa que abarca desde el diseño, recolección, y posterior análisis de los datos.

Capítulo 2

Recolección de datos en teléfonos inteligentes

A continuación se presentan soluciones de recolección de datos de código abierto utilizando equipos móviles, se describen sus características principales y sus casos de uso.

2.1. Open Data Kit

Open Data Kit (ODK) es un conjunto de herramientas de código abierto que ayuda a las organizaciones para la creación, realización y manejo de soluciones de recolección de datos móviles.

Los principales desarrolladores son investigadores del Departamento de Ingeniería y Ciencias de la Computación de la Universidad de Washington (UW) y miembros activos de Change, un grupo multidisciplinario de la UW que explora como la tecnología puede mejorar la vida de las poblaciones más desfavorecidas en el mundo.

ODK inició como un proyecto sabático de google.org bajo la dirección de Gaetano Borriello en abril del 2008 en las oficinas de Google en Seattle. Richard Anderson lidera el proyecto actualmente desde el fallecimiento de Gaetano a principios del 2015.

El proyecto es financiado por el programa Google Focused Research Award y a través de donaciones por parte de los usuarios. ODK tiene el apoyo de una creciente comunidad de desarrolladores, implementadores y usuarios [Kita].

ODK consiste de cuatro herramientas principales: Collect, Aggregate, Build.

ODK Collect es una plataforma móvil que interpreta complejas lógicas de aplicación y soporta la manipulación de tipos de datos que incluyen texto, localización, imágenes, audio, vídeo, y códigos de barra.

ODK Aggregate provee un servidor de fácil despliegue que soporta la subida de datos de las encuestas, almacenamiento, transferencia de datos en la nube y también en servidores locales.

Finalmente, ODK Build es una interfaz gráfica que permite el diseño de la lógica utilizada por las demás herramientas.

2.1.1. Diseño

Open Data Kit está diseñado como un conjunto modular de componentes que pueden ser utilizados individualmente o en varias configuraciones (incluyendo módulos que no sean de ODK) de modo a ser utilizado como un servicio de información en regiones en desarrollo.

Diseñador de formularios

De manera a permitir que los no programadores puedan construir formularios con lógicas complejas e interacciones ODK proporciona una interfaz gráfica que se puede utilizar desde un navegador web, en donde se permite a los usuarios crear el formulario con una paleta de controles. El diseñador produce la lógica necesaria para que el formulario pueda ser utilizado por las demás herramientas de manera a que pueda ser mostrado correctamente al usuario como también crear las definiciones para que pueda ser posible crear bases de datos en donde se almacenará la información recolectada.

Cliente smartphone

Aplicación de teléfonos inteligentes que permite a los usuarios descargar el formulario, interactuar con el mismo usando la interfaz táctil o con el teclado, y enviar información a los servidores inalámbricamente.

Almacenamiento en servidores

Para simplificar el proceso de almacenamiento y manejo de datos se utiliza un servidor que puede disponibilizarse localmente en una PC o en una infraestructura de computación en la nube. El servidor construye el almacenamiento de datos para cada formulario cuando el mismo es enviado, lo cual implica que el formulario tiene dos propósitos: recolectar información cuando se muestra en el dispositivo cliente, y servir de guía para que el servidor construya su correspondiente base de datos. Además, el cliente que mostrará el formulario puede obtenerlo desde el servidor y enviar los datos recolectados al mismo. Los usuarios pueden ver los resultados en formato de tabla, o exportar los datos en formatos como CSV para importarlo en herramientas de análisis estadísticos, JSON para servidores externos o KML para visualizarlo en software de mapas.

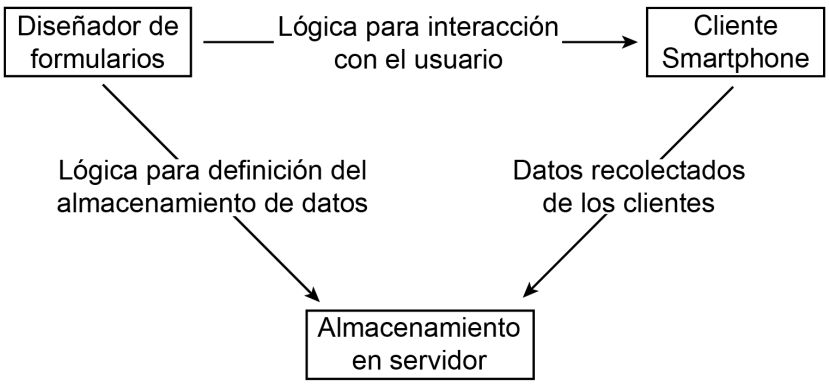


Figura 2.1: Componentes de ODK

2.1.2. Implementación

A. XForms: Un formato común

De manera a asegurar que cada herramienta se pueda utilizar independientemente pero también en conjunto se utiliza el estándar XForm [W3C] diseñado por la World Wide Web Consortium (W3C), los XForms son definiciones de formularios basados en XML.

ODK implementa el subconjunto OpenRosa [Enk] del estándar XForm que hace que las herramientas sean compatibles con otras en la comunidad y permita a las organizaciones cambiar fácilmente a otras tecnologías y sistemas sin tener que reconstruir sus aplicaciones.

B. Build: Diseñador de formularios

Gracias al uso del estándar XForm, ODK provee a los creadores con opciones flexibles para diseñar servicios a medida de sus necesidades. Desafortunadamente, esto trae un costo en cuanto a facilidad de uso. La estructura y sintaxis de los formatos basados en XML es muy compleja para describir la gran mayoría de los formularios, que generalmente son cortos y lineales. Esto a menudo lleva a confusiones entre los usuarios y representa una barrera en la adopción.

ODK Build facilita a los usuarios la creación de la lógica del formulario y la generación del XForm sin necesidad de conocer el estándar.

Build permite a los diseñadores usar una interfaz de tipo “arrastrar y soltar”, en donde coloca los controles que van a interactuar con el usuario. Esta herramienta está construida como una aplicación web HTML 5, hace uso de Javascript y Ruby Rack.

ODK Build está disponible globalmente en el sitio `http://build.opendatakit.org`, sin embargo, el código fuente puede ser descargado y correrlo localmente.

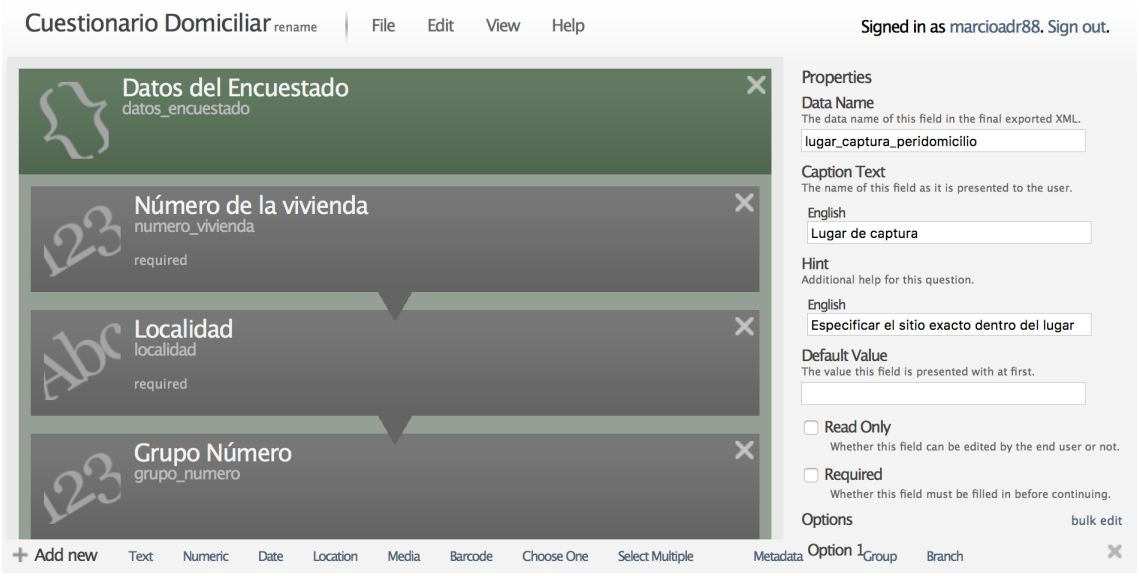


Figura 2.2: ODK Build

C. Collect: Cliente Smartphone

ODK Collect es el cliente móvil que corre en dispositivos con el sistema operativo Android, el mismo toma la definición XML del XForm y lo renderiza en la pantalla de manera a que el usuario pueda interactuar con él.

Los usuarios pueden navegar entre las preguntas moviendo el dedo entre las pantallas (similar a cambiar las páginas de un libro), o también pueden ingresar al índice de preguntas para seleccionar una en específica.

Cada pregunta corresponde a un tipo, los cuales pueden ser: texto libre, entero, decimal, selección única, selección múltiple, imagen, audio, vídeo, código de barra, o localización GPS. Cualquier pregunta puede ser de tipo solo lectura para mostrar información, en lugar de recolectarla.

Collect soporta múltiples lenguajes, validaciones avanzadas, chequeo de expresiones regulares de los datos introducidos. Adicionalmente, soporta bifurcaciones y repeticiones de grupo de preguntas permitiendo interacciones más complejas (por ejemplo, recogiendo datos particulares de cada miembro de una familia).

De manera a soportar operaciones sin conexión, Collect guarda la definición del formulario y los datos recogidos en el teléfono como archivos XML asociados con datos de tipo binario (imágenes, audio, vídeo, etc). El usuario puede optar por la sincronización con el servidor en cualquier momento usando una conexión a internet disponible. Los archivos son enviados utilizando un POST HTTP a cualquier servidor compatible con OpenRosa. Estos archivos también pueden ser transferidos desde el teléfono a una computadora con un cable USB, o extrayendo los datos de la tarjeta SD del teléfono.

Collect está escrito en Java y corre en teléfonos inteligentes con Android, un sistema operativo de código abierto. Android simplifica el desarrollo debido a su API

estandarizada que permite que los programas no tengan que ocuparse de los detalles específicos del hardware. Además, como Android no está atado a ninguna implementación de hardware en particular puede ser usado en una variedad de dispositivos incluyendo teléfono inteligentes y tabletas.

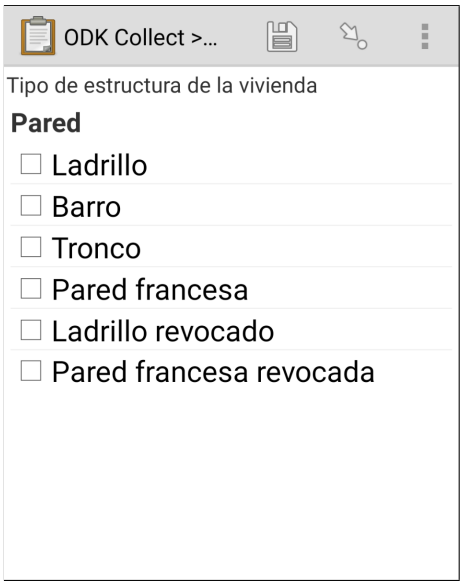


Figura 2.3: ODK Collect

D. Aggregate: Servidor de almacenamiento

Aggregate provee un repositorio para el manejo de los datos recolectados, además de interfaces estándar para extraer datos (hojas de cálculo, consultas, etc) y se integra a los sistemas existentes usando peticiones HTTP. Soporta el retorno de datos en varios formatos estándar incluyendo CSV (para análisis estadísticos), KML y JSON (para utilizarlo con servicios externos). Está diseñado para ser un almacén de datos genérico que puede ser utilizado en la plataforma de computación preferida por el usuario, pudiendo recibir datos de clientes (teléfonos y servidores) que implementen OpenRosa. Aggregate está escrito en Java y puede ser desplegado en cualquier contenedor web Java.

Su interfaz está diseñada para que sea de fácil uso inclusive para personas con conocimiento básico de computadoras, los usuarios simplemente necesitan enviar su XForm y la plataforma automáticamente crea el almacenamiento de datos basado en la definición del formulario. Además, se brinda la posibilidad de construir consultas usando la interfaz web, pudiendo crear reportes sin necesidad de conocer detalles de la implementación de la base de datos [Har+10].

Figura 2.4: ODK Aggregate: Visualización de localizaciones recolectadas en el mapa

2.1.3. Futuro

Actualmente se está trabajando en la versión 2.0 de ODK, el cual incluye mejoras sobre la versión 1.0 (la última estable) que se analizó anteriormente en esta sección. La versión 2.0 aún está en un estado de Beta, pero se pueden destacar las siguientes mejoras:

- Nueva implementación del renderizado de formularios: el diseñador de encuestas podrá especificar la disposición de las preguntas en el formulario.
- Navegación más flexible en el formulario: se podrá especificar la navegación entre las preguntas, no necesariamente será secuencial como en la versión actual.
- Mejoras en los grupos de repetición: se dispondrán de controles para abrir y editar otros formularios con enlaces para regresar a la encuesta origen. Estos enlaces pueden ser dos formas: formularios anidados, o relaciones arbitrarias entre formularios [Kitb].

Open Data Kit está liderado por la Universidad de Washington, pero se encuentra en un proceso de transición desde un proyecto de investigación a un proyecto autosostenible por la comunidad de código abierto con la propiedad de una fundación sin fines de lucro [Kitc].

2.1.4. Proyectos relacionados

La naturaleza de código abierto y la utilización de estándares ha provocado la creación proyectos que enriquecen el ecosistema de Open Data Kit, a continuación, describiremos a los más representativos.

Enketo Smart Paper

Enketo renderiza formularios en el navegador web gracias a la utilización del estándar XForm y la utilización de OpenRosa. Ofrece la implementación de todos los tipos de preguntas incluyendo soporte de trabajo sin conexión, lógica de bifurcación de formularios y adaptabilidad a navegadores móviles.

Además, ofrece la posibilidad de conectar con ODK Aggregate para el repositorio de datos, pudiendo de esa manera desplegar encuestas masivas en internet [Pap].

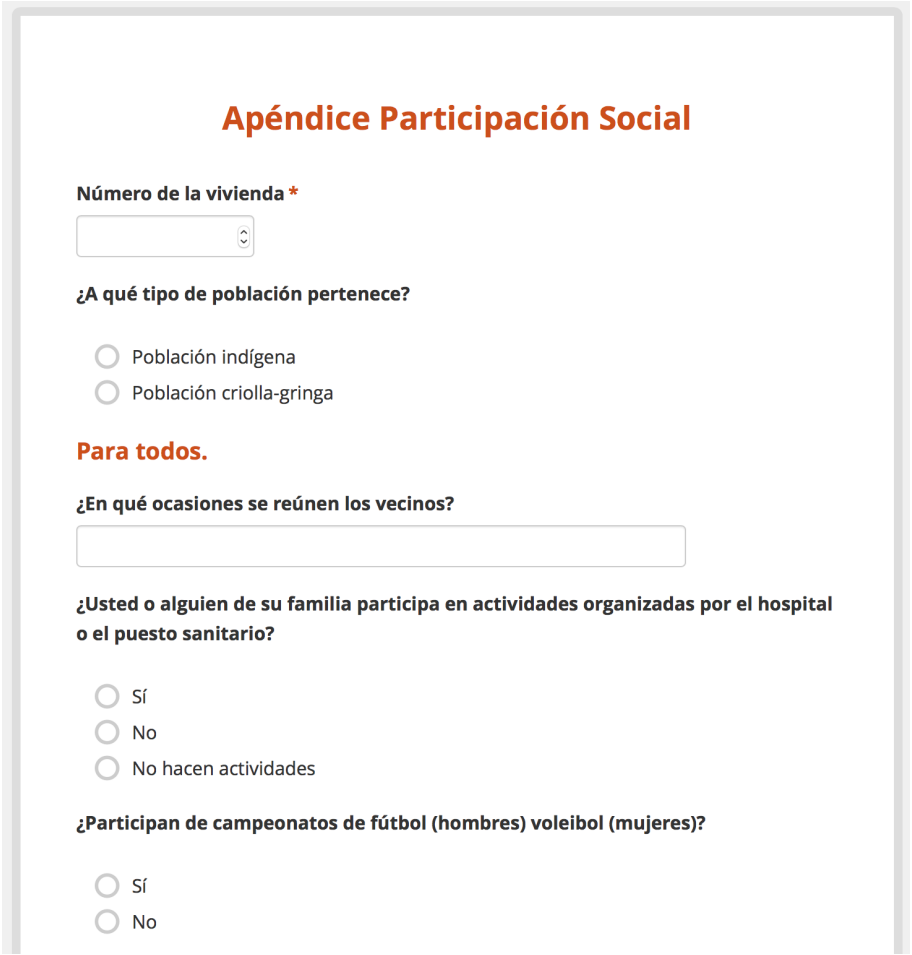


Figura 2.5: Formulario web usando Enketo

Formhub

Formhub es un software de código abierto que permite a los usuarios, especialmente los no técnicos, la gestión de los esfuerzos que demanda una recolección de

datos móviles. Fue desarrollado por el Laboratorio de Ingeniería Sostenible de la Universidad de Columbia.

Provee una interfaz web en donde los mismos pueden añadir sus formularios y ver los datos que fueron recogidos mediante una interfaz similar a ODK Aggregate. Gracias a la utilización de estándares, es compatible con ODK Collect para la recolección de datos móviles, y proporciona integración con Enketo Smart Paper para el despliegue de encuestas en línea.

De esta manera, Formhub provee una solución integral que está listo para utilizarse para por cualquier organización sin incurrir en esfuerzos técnicos que implica la puesta a punto de servidores y los costes monetarios asociados a los mismos [For].

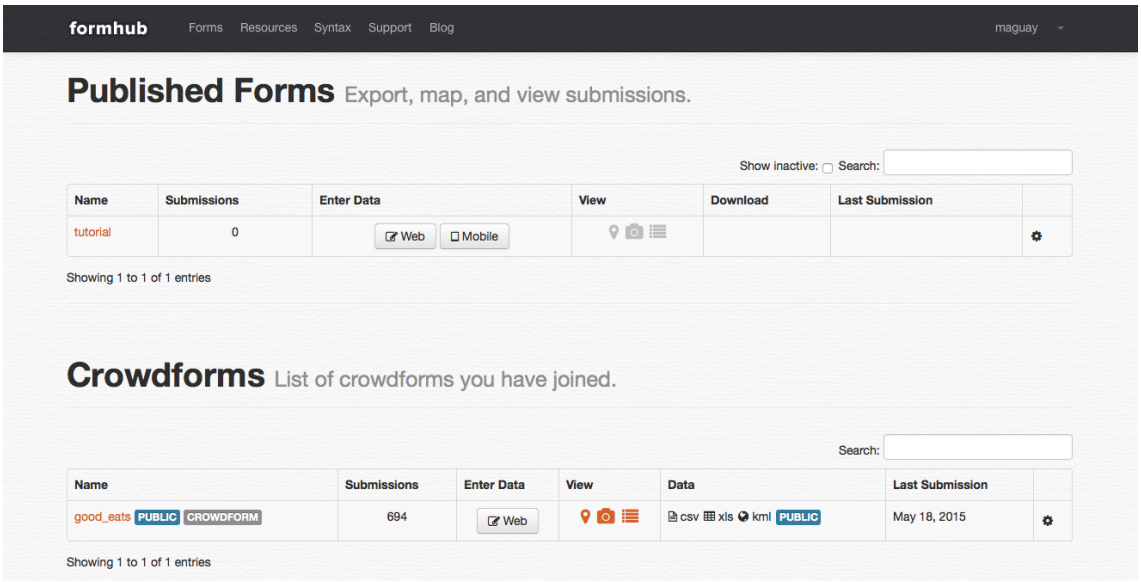


Figura 2.6: Interfaz web de Formhub

KoBo Toolbox

Es una suite de código abierto de herramientas para la recopilación y análisis de datos humanitarios, especialmente en lugares remotos o después de un desastre. Se compone de tres herramientas diferentes, uno para el diseño de formularios, uno para la recolección de datos y otro para el análisis de datos. KoBo Toolbox puede utilizarse desde su sitio web o instalado en un servidor local. Utiliza XForms para la definición de los formularios, que pueden ser creados mediante una interfaz similar a ODK Build, además, para la recolección de datos móviles se utiliza KoBo Collect, que está basada en ODK Collect y también se apoya en Enketo Smart Paper para las encuestas web.

Fue diseñado por la Iniciativa Humanitaria de Harvard con el apoyo de las Naciones Unidas, el Comité Internacional de Rescate y el gobierno de los Estados Unidos [Ini].

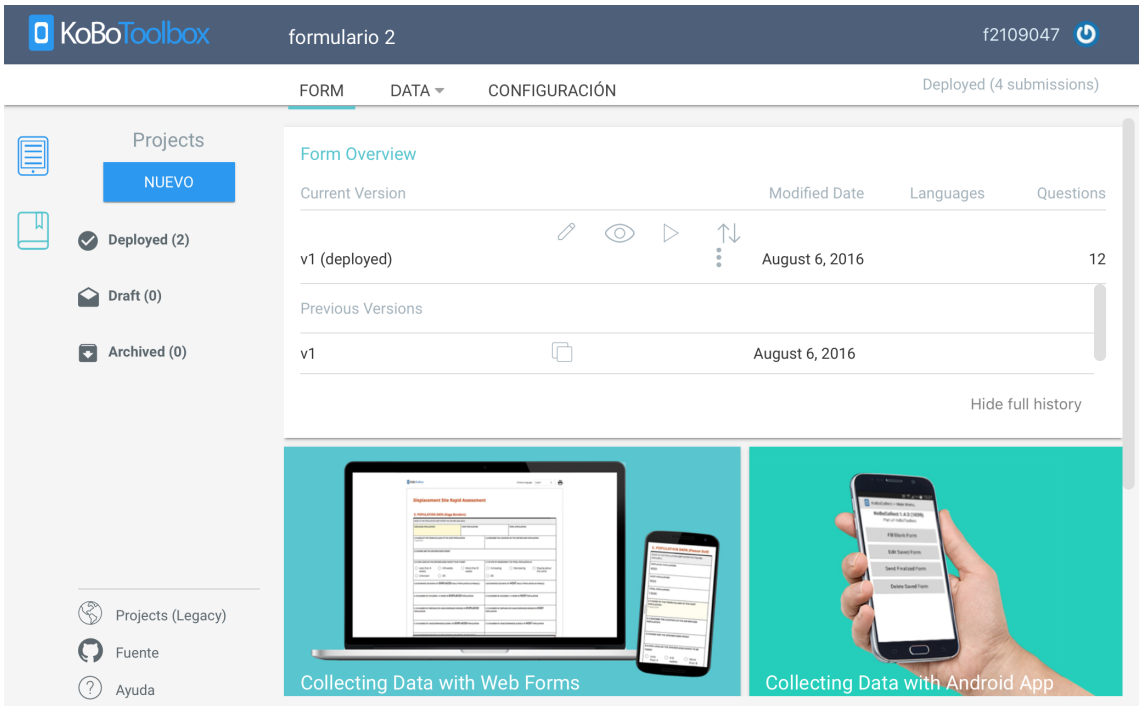


Figura 2.7: Interfaz web de KoBo Toolbox

2.1.5. Casos de éxito

Enfoque tecnológico innovador para la respuesta al brote de epidemia de la enfermedad del virus del Ébola utilizando tecnologías de Open Data Kity Form Hub

El reciente brote del Virus de Ébola (EVD) en el oeste de África se ha llevado muchas vidas. La contención efectiva de este brote se basa en la coordinación y la comunicación inmediata y efectiva a través de diversas intervenciones, detección temprana y la pronta respuesta con críticos para un control exitoso.

Debido a la necesidad de la obtención de datos y comunicación de forma rápida para el descubrimiento temprano de nuevos casos de EVD para la toma adecuada de decisiones, se hizo imperativo brindar a los miembros del equipo de respuesta con tecnologías y soluciones que permitan el flujo de datos rápido y confiable.

Se utilizaron Open Data Kit y Form Hub en combinación con ArcGIS. Se notó una importante mejora en el reporte diario y el seguimiento de los contactos luego del despliegue de las tecnologías integradas. El tiempo entre la identificación de los contactos sintomáticos y la evacuación a las instalaciones de aislamiento, y además el tiempo para la recepción de los resultados laboratoriales fueron reducidos, y la toma de decisiones informadas pudieron tomarse entre todos los involucrados.

El uso de tecnologías innovadoras en la respuesta del brote EVD en Nigeria contribuyó significativamente al control del brote y la contención de la enfermedad, proporcionando una plataforma valiosa para alertas tempranas y guiando acciones informadas [Tom+15].

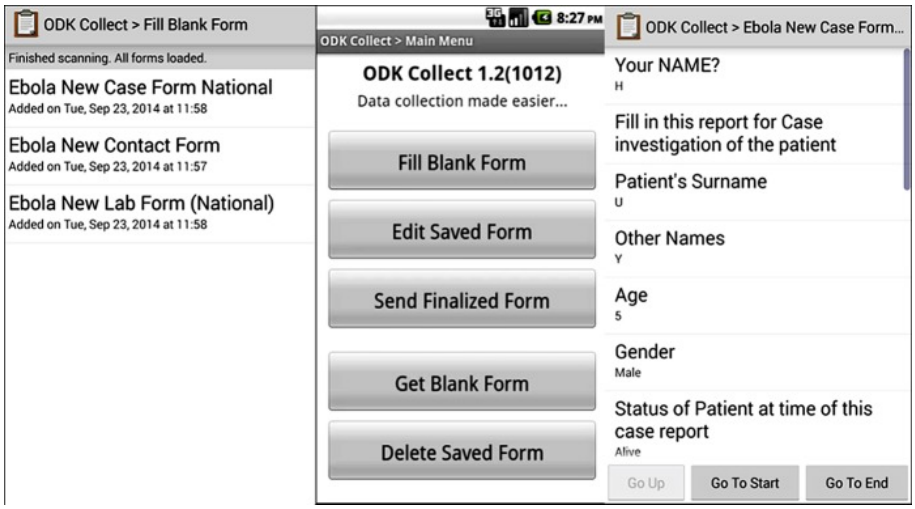


Figura 2.8: Capturas de pantalla de la aplicación ODK Collect con los formularios utilizados para el control del Ébola

El código abierto y el papel de los datos abiertos en el auxilio del terremoto de Nepal

Un devastador terremoto de magnitud 7.8 golpeó Nepal el 25 de abril del 2015, matando más de 9000 personas, hiriendo a un millar más y dejando a 3 millones sin un hogar.

Inmediatamente luego del terremoto, el gobierno, fuerzas de seguridad local e internacional, y las agencias de asistencia internacional se pusieron a tratar de ayudar. Sin embargo, había una falta de coordinación entre esos grupos.

Utilizando OpenStreetMap los miembros de la comunidad trabajaron en crear un mapa detallado de las áreas afectadas, el cual se utilizó para planear y movilizar los recursos.

Desde que ocurrió el terremoto se empezaron a desarrollar varios proyectos de recolección de datos para medir el impacto del daño causado, seguimiento de las asistencias, y monitoreo de las reconstrucciones.

Se utiliza KLL Collect, una aplicación basada en Open Data Kit, para el proyecto de recolección de datos. Construir la plataforma sobre ODK permitió añadir las características necesarias y personalizar la aplicación con las necesidades del proyecto. Se utilizaron las tecnologías de servidor de ODK y se centraron los esfuerzos en adaptar la aplicación móvil. La activa comunidad y los recursos en línea existentes hicieron que obtener ayuda cuando se encontraba algún problema fue de mucha ayuda [Pud16].

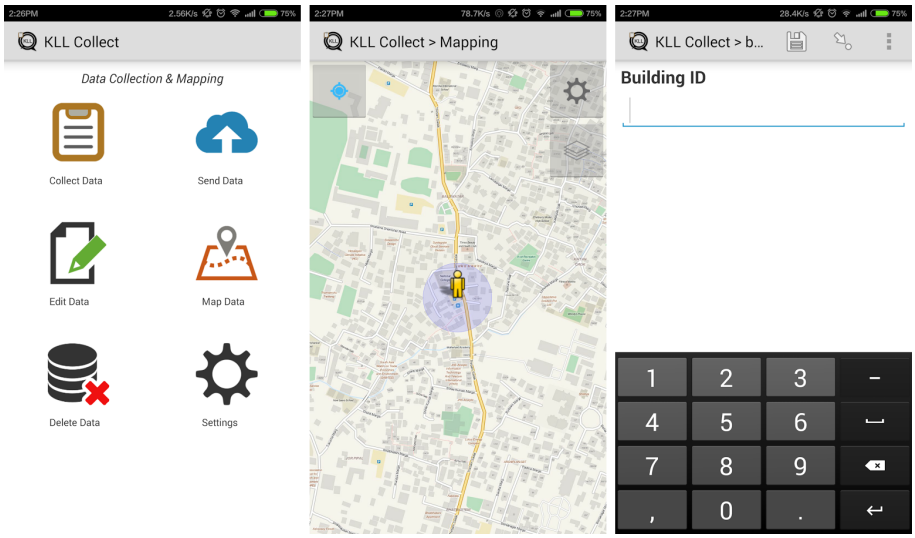


Figura 2.9: KLL Collect

Reducción de errores y retrasos en recolección de millones de puntos de datos del Programa Mundial de Alimentos (World Food Programme)

El proyecto Purchase for Progress (P4P) utiliza el poder adquisitivo del World Food Programme (WFP) junto con técnicas inventivas locales y las mejores prácticas, para conducir a pequeños productores a cadenas de valor formales donde ellos podrán ganar más dinero. El proyecto inició en el año 2008 y ha mejorado la vida de cientos de miles de agricultores en 20 países en África, Asia y Centroamérica.

El monitoreo y la evaluación son claves para demostrar la eficacia del P4P, por lo que el WFP recopila datos longitudinales sobre los hogares y las organizaciones de agricultores para el seguimiento del proyecto. En 2011, el WFP se asoció con AERC para recoger, limpiar, analizar, gestionar e informar sobre los datos cuantitativos generados por 17 de los 20 países P4P.

Al inicio del proyecto, los conjuntos de datos, cada uno con alrededor de 1600 variables, se recogieron en papel. En una fecha posterior, los datos se introducen manualmente en las computadoras. Este proceso basado en papel dio lugar a errores en la recolección de datos, largas demoras antes del procesamiento de datos, y dificultades en el seguimiento de las encuestas en curso desde las oficinas AERC en Nairobi. Fue una pesadilla de gestión de datos.

En 2013, con el apoyo de Nafundi, AERC comenzó a utilizar ODK para la recolección de datos en Ruanda. Los beneficios fueron inmediatos. El tiempo entre la recolección y análisis de datos se redujo casi a la mitad. Además, el seguimiento remoto en tiempo casi real permitió la captura y la corrección de errores mientras que los encuestadores estaban aún en el campo.

Desde entonces AERC ha utilizado ODK para recoger datos P4P en Kenia, Etiopía, El Salvador, Ghana y Zambia. Hasta la fecha, más de 2300 hogares y 230 organizaciones de agricultores han sido encuestados utilizando ODK en estos países. Esto representa más de 3,6 millones de puntos de datos recogidos con ODK.

Con cada despliegue, ha habido reducciones drásticas en el tiempo y esfuerzo

necesarios para la recogida y análisis de datos. También se ha visto una enorme mejora en la calidad de los datos. Por estas razones, AERC seguirá utilizando ODK para la recolección y análisis de datos [Bru15].

2.2. EpiInfo

Epi Info es una suite de dominio público de herramientas de software interoperables diseñadas para la comunidad global de profesionales de la salud pública y los investigadores. Contempla una forma fácil de entrada de datos y la construcción de bases de datos, una experiencia personalizada de entrada de datos, y análisis de datos con estadísticas epidemiológicas, mapas y gráficas para profesionales de la salud pública que pueden carecer de un soporte de tecnología de la información. Epi Info se utiliza para la investigación de brotes; para el desarrollo de pequeños a medianos sistemas de vigilancia de enfermedades; también como componente de análisis, visualización y presentación de informes de sistemas más grandes; y en la formación continua en la ciencia de la epidemiología y métodos analíticos de la salud pública en las escuelas de salud pública en todo el mundo. [Infa]

El software es gratuito y está disponible para su descarga públicamente en el sitio web de Epi Info. Está diseñado para correr en sistemas operativos Microsoft Windows, a partir de la versión XP.

2.2.1. Diseño

Está compuesto de cuatro herramientas principales utilizadas para recolectar, analizar o visualizar datos:

- Crear formularios: Crear cuestionarios usando uno o más formularios para recolectar o ver datos.
- Ingresar datos: Ingresar datos y visualizar registros existentes.
- Analizar datos (Panel Visual y Vista Clásica): Administrar datos, ejecutar análisis estadísticos, y generar listas, tablas y gráficos.
- Crear mapas: Crear mapas desde un servidor de mapas, archivos KML o archivos de formas.

Figura 2.10: Epi Info: Herramientas disponibles en el programa

Diseñador de formularios

El módulo de diseñador de formularios permite a los usuarios crear cuestionarios y formularios de entrada de datos. Los usuarios pueden ubicar preguntas y campos de entrada de datos en una o varias páginas y adaptar el proceso de entrada de datos con patrones de saltos condicionales, validación de datos y cálculos personalizados programados por el usuario.

Entrada de datos

El módulo de entrada de datos crea automáticamente la base de datos del cuestionario diseñado con el diseñador de formularios. Ahí los usuarios pueden ingresar nuevos datos, modificar datos existentes o buscar registros. En este módulo los formularios son desplegados y los usuarios realizan la entrada de datos mientras que los datos son validados o cualquier cálculo automático especificado en Diseñador de Formularios se realiza.

Análisis

El módulo de análisis se utiliza para leer y analizar datos ingresados con el módulo de entrada de datos o datos importados de 24 formatos diferentes de datos. Estadísticas epidemiológicas, tablas, gráficos y mapas son producidos con comandos simples para el usuario. Cuando cada comando es ejecutado, es guardado en el editor del programa donde podrá se personalizado y guardado, compartido y utilizado en el futuro a medida que los datos son modificados.

Mapas

El módulo de mapas muestra mapas geográficos con datos de Epi Info. El módulo fue construido sobre el software MapObjects del Instituto de Investigación de Sistemas Ambientales (ESRI). Permite visualizar archivos de forma que contienen los límites geográficos cruzados con los datos resultantes del módulo de análisis.

2.2.2. Implementación

A. Diseñador de formularios

El diseñador de formularios es la herramienta utilizada para diseñar la encuesta, formulario o cuestionario, adaptar el proceso de entrada de datos y especificar la secuencia de las páginas o pestañas. También es donde se personalizan las validaciones de datos que se deseen realizar cuando se ingresan los datos. La recolección de datos en Epi Info está organizada en proyectos. Cada proyecto puede tener uno o más formularios los cuales pueden a su vez tener una o varias páginas. En cada página, uno o más campos de entrada de datos son añadidos para recolectar elementos de datos individuales.

Los campos añadidos a una página pueden ser cualquiera de una variedad de tipos de campos correspondientes al tipo de dato necesitado y los tipos de análisis que pueden ser anticipados. Normalmente se utiliza el formato de base de datos de Microsoft Access para el guardado de los datos, a no ser que sea especificada en la configuración una conexión a una base de datos de Microsoft SQL Server, en cuyo caso se utiliza el formato por defecto para SQL Server.

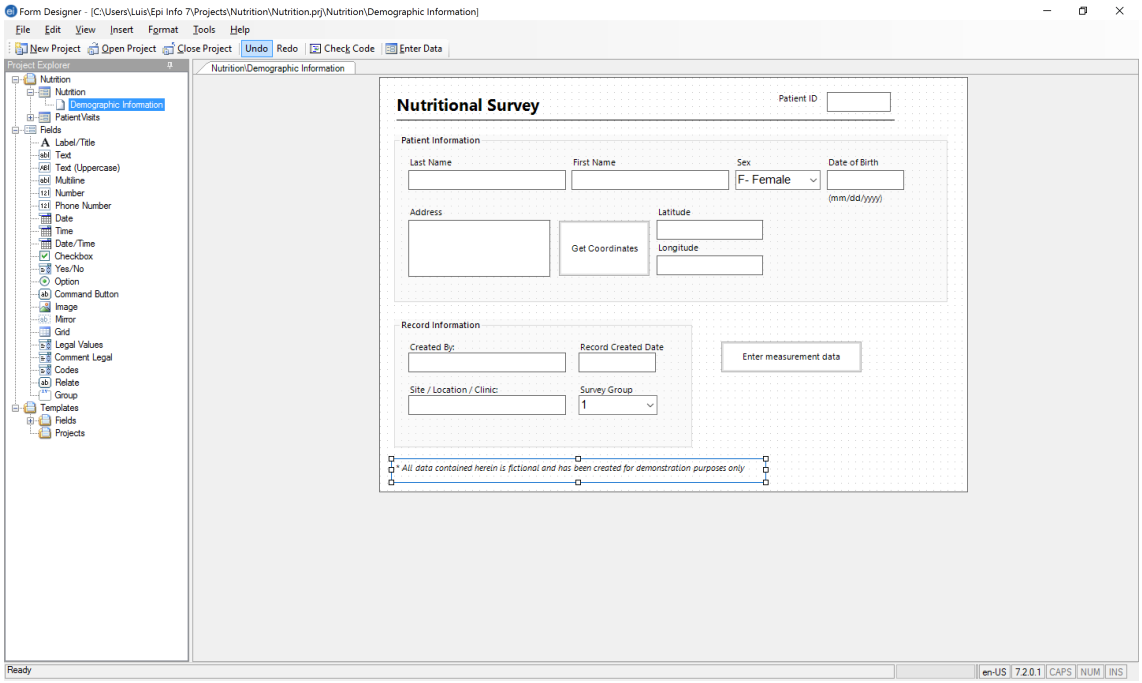


Figura 2.11: Epi Info: Pantalla del diseñador de formularios

B. Entrada de datos

El módulo de entrada de datos permite a los usuarios ingresar los datos en los formularios diseñados en el diseñador de formularios. La información ingresada en el formulario es almacenada en tablas de datos de Microsoft Access o Microsoft SQL Server según se haya seleccionado en la configuración. En este módulo se pueden ingresar nuevos datos, modificar datos existentes o buscar registros ingresados.

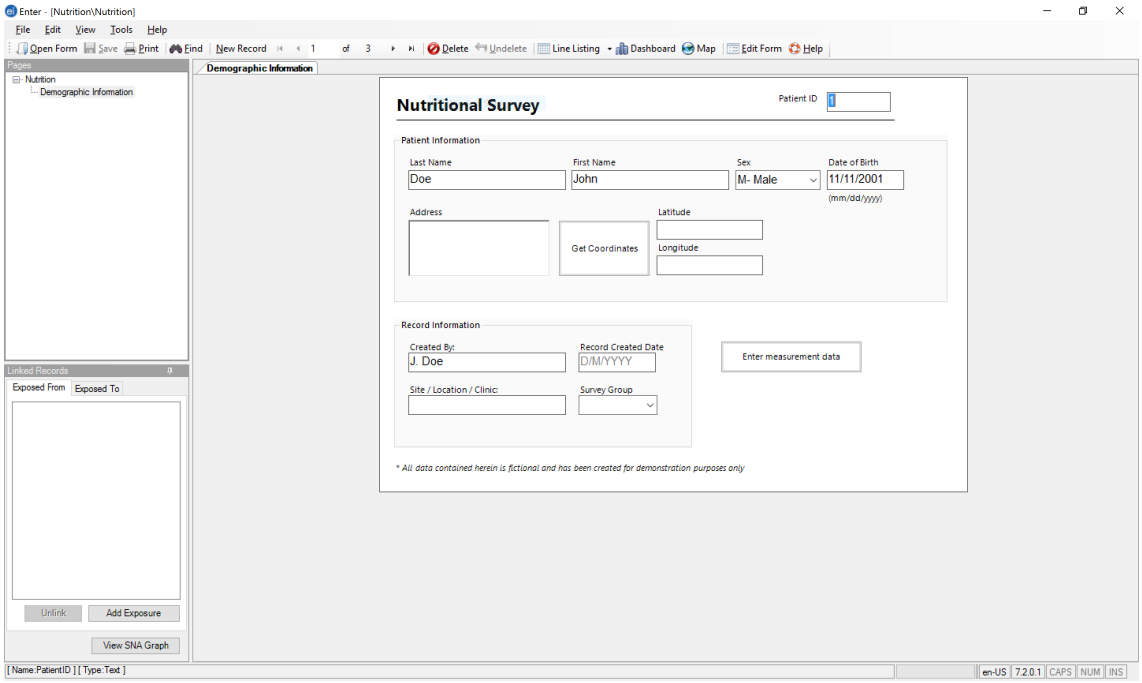
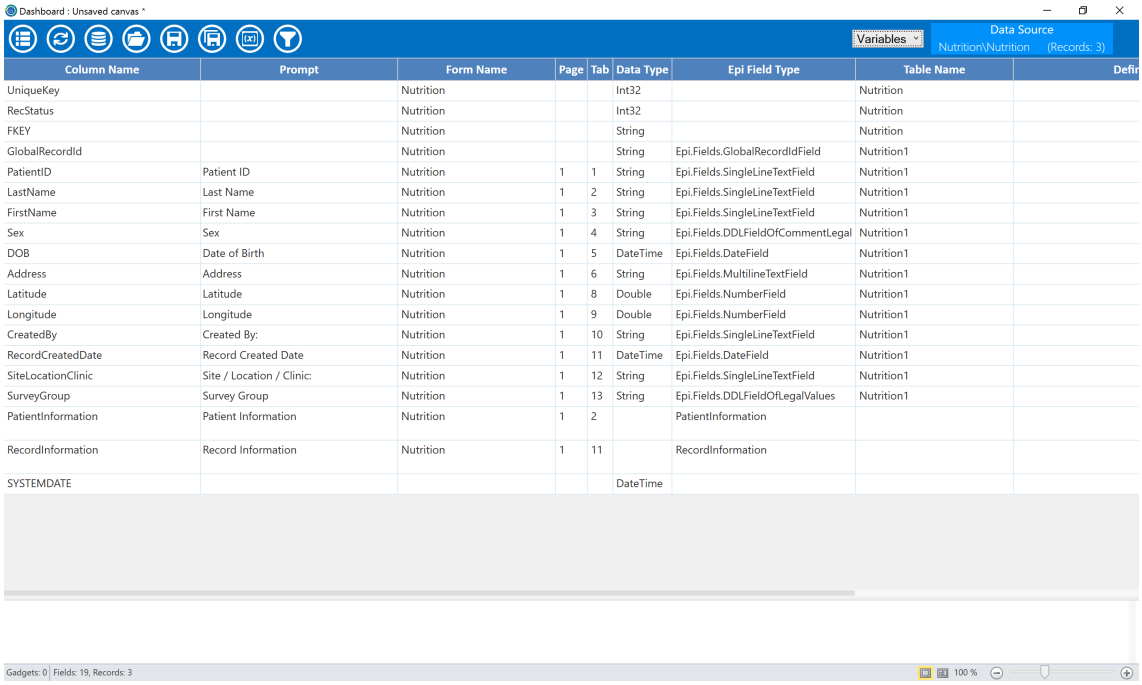


Figura 2.12: Epi Info: Pantalla de entrada de datos

C. Análisis

El análisis de datos tiene dos alternativas: la versión clásica que analiza los datos a través de comandos predefinidos y el panel de visualización. El panel de visualización está diseñado para ser intuitivo y simple de usar. Con el uso de interfaz gráfica y gadgets, la necesidad de escribir código de programación es minimizada. Los datos pueden ser seleccionados, ordenados, listados o manipulados con varios gadgets del panel de visualización. Las herramientas de análisis estadísticos disponibles incluyen frecuencias, medianas y más procesos de cálculos estadísticos avanzados (como por ejemplo regresiones lineares y regresiones logísticas). También posee funcionalidad de generación de gráficos estadísticos como gráficos de barras, entre otros.



Column Name	Prompt	Form Name	Page	Tab	Data Type	Epi Field Type	Table Name	Defir
UniqueKey		Nutrition			Int32		Nutrition	
RecStatus		Nutrition			Int32		Nutrition	
FKEY		Nutrition			String		Nutrition	
GlobalRecordId		Nutrition			String	Epi.Fields.GlobalRecordIdField	Nutrition1	
PatientID	Patient ID	Nutrition	1	1	String	Epi.Fields.SingleLineTextField	Nutrition1	
LastName	Last Name	Nutrition	1	2	String	Epi.Fields.SingleLineTextField	Nutrition1	
FirstName	First Name	Nutrition	1	3	String	Epi.Fields.SingleLineTextField	Nutrition1	
Sex	Sex	Nutrition	1	4	String	Epi.Fields.DDLFieldOfCommentLegal	Nutrition1	
DOB	Date of Birth	Nutrition	1	5	DateTime	Epi.Fields.DateField	Nutrition1	
Address	Address	Nutrition	1	6	String	Epi.Fields.MultilineTextField	Nutrition1	
Latitude	Latitude	Nutrition	1	8	Double	Epi.Fields.NumberField	Nutrition1	
Longitude	Longitude	Nutrition	1	9	Double	Epi.Fields.NumberField	Nutrition1	
CreatedBy	Created By:	Nutrition	1	10	String	Epi.Fields.SingleLineTextField	Nutrition1	
RecordCreatedDate	Record Created Date	Nutrition	1	11	DateTime	Epi.Fields.DateField	Nutrition1	
SiteLocationClinic	Site / Location / Clinic:	Nutrition	1	12	String	Epi.Fields.SingleLineTextField	Nutrition1	
SurveyGroup	Survey Group	Nutrition	1	13	String	Epi.Fields.DDLFieldOfLegalValues	Nutrition1	
PatientInformation	Patient Information	Nutrition	1	2		PatientInformation		
RecordInformation	Record Information	Nutrition	1	11		RecordInformation		
SYSTEMDATE					DateTime			

Figura 2.13: Epi Info: Pantalla de análisis de datos con la lista de variables disponibles

D. Mapas

La herramienta de mapas posee la característica de visualizar múltiples vistas del mismo conjunto de datos. Los mismos pueden ser filtrados o mostrados a través de una serie de tiempo usando características de la herramienta. La información mostrada en la pantalla principal del mapa en forma de capas. Las capas de datos existen en forma de grupos de casos, mapas coropléticos o mapas de densidad de puntos. Las capas de referencia añaden los límites geográficos y marcadores desde archivos de formas, servidores de mapas o archivos KML.

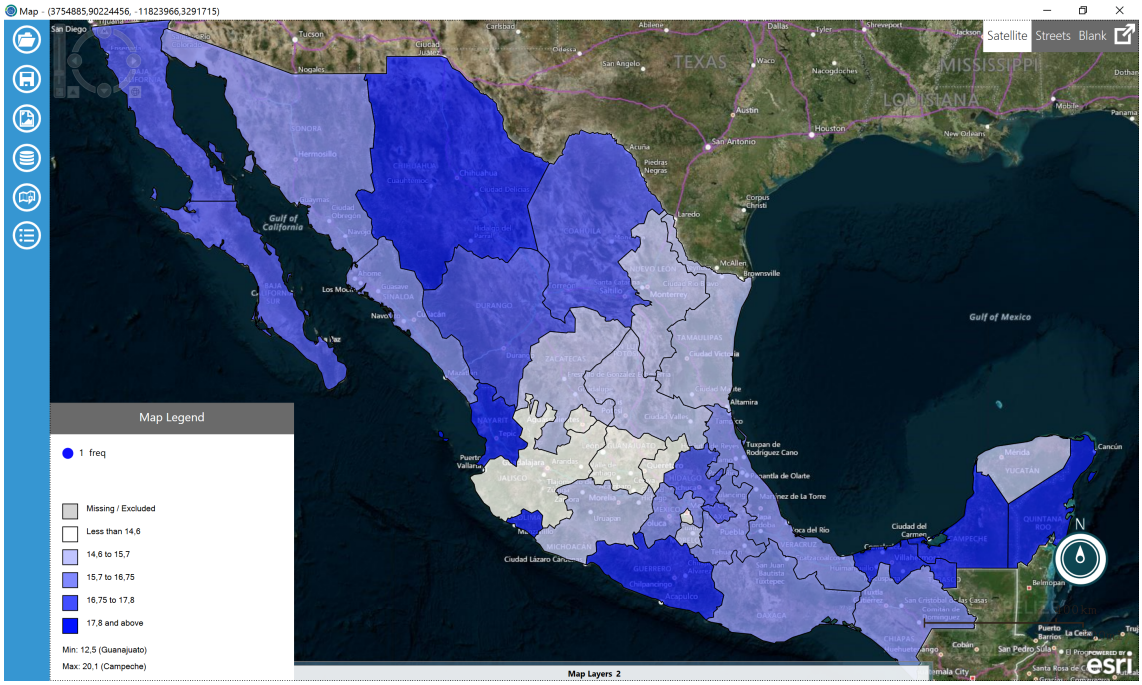


Figura 2.14: Epi Info: Vista de mapa coroplético

2.2.3. Futuro

Epi Info continúa desarrollándose como software libre, el código fuente está disponible en el sitio web Codeplex [Infb]. Ingresos de datos vía web, análisis vía web y herramientas de recolección móviles en Android y iOS están disponibles para su utilización. Ambas versiones móviles funcionan como herramientas de recolección de datos que luego se sincronizan con el software Epi info en una PC con Windows directamente a través de un cable USB o sincronización a través de la web utilizando servicios de Microsoft Azure.

Figura 2.15: Epi Info: Aplicación para Android

2.2.4. Casos de éxito

Contaminación por parásitos caninos de importancia zoonótica en playas de la ciudad de Corrientes, Argentina

El objetivo del estudio fue evaluar la presencia de parásitos caninos, capaces de infectar al hombre, en playas de la ciudad de Corrientes. Durante el año 2001, se investigaron varias de playas de la ciudad, donde se obtuvieron y analizaron dos tipos de muestras: materia fecal canina y arena.

Para el análisis estadísticos de los datos de las muestras de arena se utilizó el test de comparación de proporciones con el software Epi info versión 6.04.

Los resultados de las muestras realizadas indicaron que la contaminación general, tanto de materia fecal como de arena es alta y las cuatro especies de parásitos presentes son agentes etiológicos de distintas patologías en el hombre. [MFO02]

Tabla 1. Contaminación de materia fecal canina en playas de la ciudad de Corrientes, Argentina

Playa	Muestras		
	Analizadas	Contaminadas	
		N	(%)
Perichón	20	14	70
Molina Punta	13	9	69,2
Yacaré	26	10	38,5
Mitre	25	13	52
Islas Malvinas	23	12	52,2
Arazatí	16	15	93,7
Total	123	73	59,3

Figura 2.16: Resultados del estudio de la muestra de materias fecales caninas

Tabla 2. Contaminación por *Ancylostoma* spp. y *Toxocara canis* en muestras de arena de playas de la ciudad de Corrientes, Argentina

Playa (superficie en m²)	Muestras		
	Analizadas	Contaminadas	
		N°	%
Perichón (3.800)	62	23	37,1
Molina Punta (2.080)	36	7	19,4
Yacaré (5.412)	84	23	27,4
Mitre (1.088)	21	10	47,6
Islas Malvinas (3.146)	50	12	24
Arazatí (4.600)	71	31	43,6*
Total (20.126)	324	106	32,7

*Una muestra presentó huevos de *T. canis*.

Figura 2.17: Resultados de la muestra de arena de las playas de la ciudad de Corrientes

2.3. EpiCollect

EpiCollect es una plataforma de código abierto que se utiliza para la generación de proyectos de recolección de datos móviles. Consta de una aplicación web para el diseño de formularios que luego pueden ser desplegados a teléfonos inteligentes compatibles.

El nombre Epi Collect se debe a que se originalmente el proyecto se utilizó para recoger datos epidemiológicos. Fue desarrollado por el Departamento de Epidemiología de Enfermedades Infecciosas, del Imperial College de Londres y financiado por Wellcome Trust [Epia].

2.3.1. Diseño

EpiCollect se divide en dos componentes principales que se complementan e interactúan para llevar a cabo el despliegue de encuestas móviles.

1. Servidor

Tiene tres funciones principales:

- Diseñar la encuesta: Provee una interfaz intuitiva en donde el usuario puede arrastrar y soltar los controles para construir el formulario. Se pueden utilizar elementos como entradas de texto, selecciones simples y múltiples. Una vez diseñado, el servidor lo disponibiliza mediante un nombre único elegido por el usuario mediante el cual se identificará el formulario.
- Almacenar los datos recogidos: Se crean estructuras de acuerdo a la definición de los formularios creados, el cual será utilizado como repositorio de los datos enviados por los clientes.
- Visualización de datos: Una vez recogidos los datos, los mismos se pueden visualizar de manera tabular, mostrando además las imágenes tomadas durante el desarrollo de la encuesta. Para los datos de tipo GPS el servidor despliega un mapa con los puntos que fueron recolectados durante la encuesta.

2. Cliente Smartphone

Aplicación de teléfonos inteligentes que permite descargar la definición del formulario desde el servidor y mostrarlo adecuadamente para la recolección de datos. Además, permite visualizar la información recogida y enviarlos al servidor una vez que se tenga una conexión a internet.

2.3.2. Implementación

- **Servidor**: Se trata de una aplicación web implementada en Python y Javascript que sirve de soporte para todo el proceso de recolección de datos. En la terminología de EpiCollect se denomina "Proyecto" al conjunto de elementos que permite llevar a cabo el despliegue de la recolección de datos móviles, el cual comprende a la definición del formulario, el almacenamiento de los datos, y su posterior visualización. El servidor es el encargado de crear proyectos los de recolección de datos, al cual el usuario deberá dar un nombre único.

En la primera etapa del proyecto, el servidor brida la posibilidad de diseñar los formularios utilizando una interfaz web, en donde el usuario dispone de una paleta de controles que le permite crear la encuesta. Una vez finalizada la etapa de diseño, se disponibiliza la definición del formulario a través del nombre del proyecto de manera a que los clientes móviles puedan obtenerlo. EpiCollect utiliza el estándar XForm para la especificación del formulario, pero implementando solo los elementos como entradas de texto, selecciones simples y múltiples. Los datos de geolocalización e imágenes no están incluidos en la definición del formulario, sino que están incluidos por defecto en todas las encuestas.

El servidor además crea un repositorio de datos para cada formulario, recibiendo y almacenando los datos enviados por los dispositivos móviles, incluyendo los datos que forman parte del formulario y los metadatos que comprenden la geolocalización y las imágenes. Finalmente, es capaz de mostrar a los usuarios los datos recogidos de manera tabular, permitiendo además la edición o eliminación de los mismos a los administradores. Los datos de geolocalización se muestran utilizando Google Maps y Google Earth, permitiendo visualizar los datos recogidos en un punto en particular.

EpiCollect disponibiliza su sitio web para desplegar encuestas online sin necesidad de incurrir en configuraciones ni requerir de personal especializado para el efecto; sin embargo, también provee la flexibilidad de instalarse en un servidor local en caso de ser necesario.

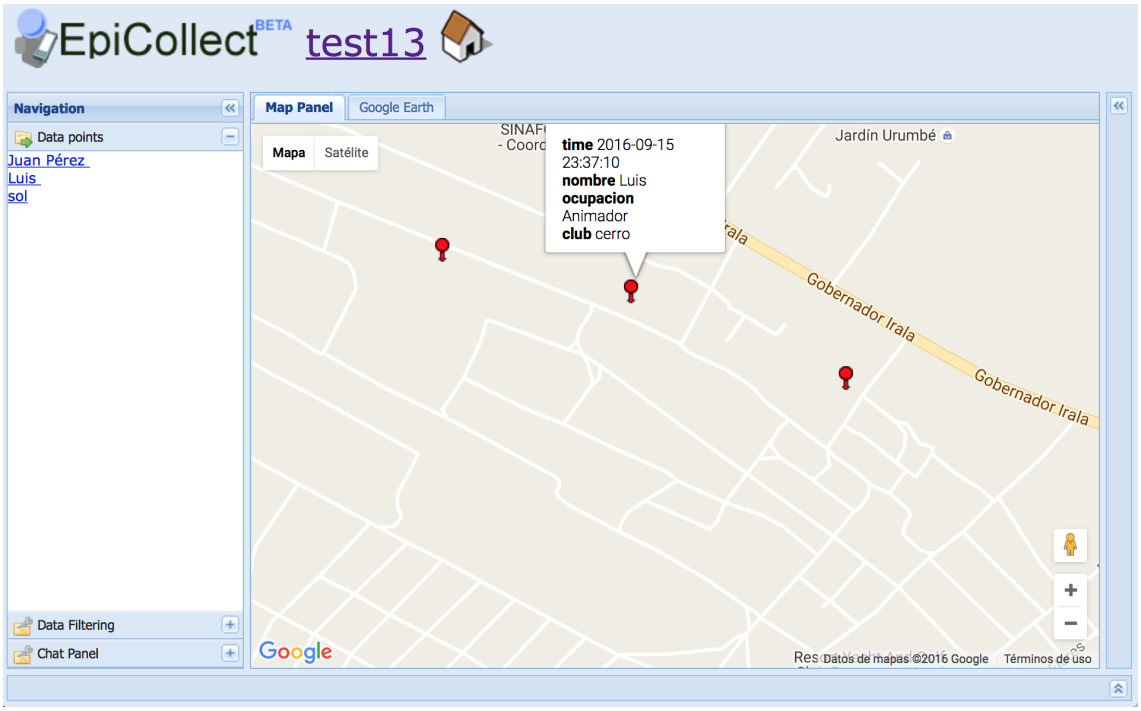


Figura 2.18: EpiCollect: Pantalla de visualización de geolocalizaciones recogidas

- **Cliente Smartphone**: Consiste en aplicaciones nativas para teléfonos móviles y tabletas con sistemas operativos Android e iOS que toman la definición de un proyecto EpiCollect a través del nombre y la dirección URL del servidor. Una vez obtenida la definición del formulario se puede iniciar el proceso de recolección de datos de acuerdo a la lógica definida, agregando además imágenes y geolocalización. La interfaz de toma de datos se presenta en una sola pantalla y las preguntas aparecen en forma de lista vertical.

El proceso de toma de datos se puede realizar sin disponer de una conexión a internet, los datos se almacenan en el dispositivo hasta que el usuario decida sincronizarlos con el servidor. Los datos recolectados se pueden consultar en cualquier momento y visualizarlos en un mapa.

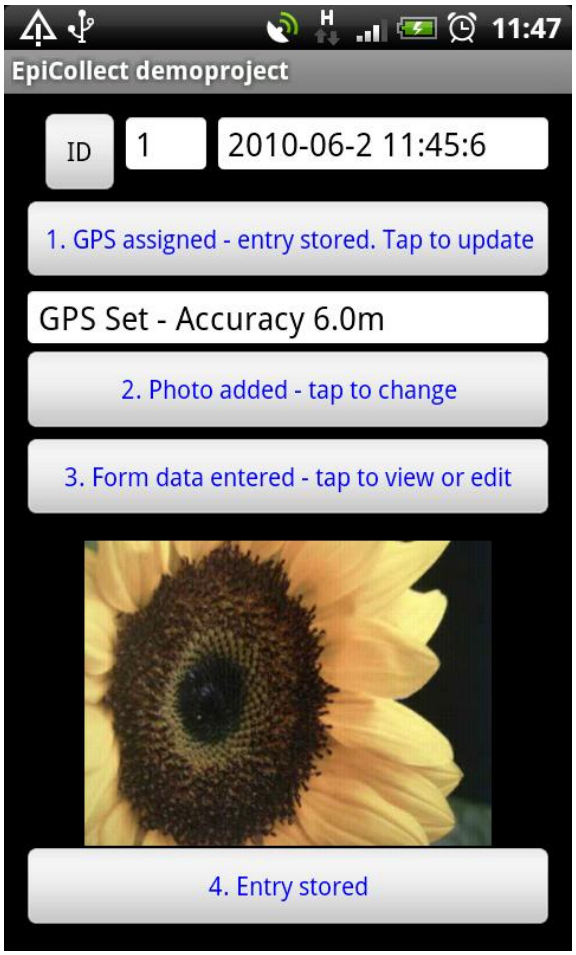


Figura 2.19: EpiCollect: Aplicación móvil de recolección de datos

2.3.3. Futuro

El Imperial College London se encuentra trabajando en EpiCollect+ el cual representan varias mejoras sobre la versión actual, actualmente se encuentra en prueba con versiones Beta. EpiCollect+ presenta las siguientes características [Epib]:

- Agrega nuevos tipos de preguntas: Fecha, hora, imágenes, localización, vídeo y escaneo de códigos de barra. En la versión anterior las imágenes y las localizaciones GPS no formaban parte de la definición del formulario, sino que se agregaban como metadatos al mismo.
- Incorporación de lógica de bifurcación del formulario, así como validaciones de las entradas.
- Los formularios se pueden vincular entre sí de manera jerárquica mediante bifurcaciones o de manera anidada.
- Se utiliza un nuevo esquema de definición de formularios denominado ecML: EpiCollect Markup Language, el cual está basado en XML.

- Nueva aplicación móvil de recolección de datos compatible con las nuevas definiciones de formularios y tipos de pregunta. Actualmente está disponible solamente para Android.

2.3.4. Casos de éxito

En septiembre del año 2009 se llevó a cabo una encuesta ilustrativa de EpiCollect en donde un grupo de personas recogieron datos en toda Europa. El trabajo de campo consistía en identificar los animales muertos que fueron infectados con un patógeno bacterial específico. Primero los encuestadores realizaban pruebas en campo para determinar si un animal estaba infectado, luego, las muestras infectadas se remitían al laboratorio para caracterizar la cepa (en este caso determinando el serogrupo de la cepa y su susceptibilidad o resistencia a un antibiótico).

Gracias a que las localizaciones fueron recolectadas se pudieron desplegar mapas con capas para identificar visualmente las zonas en donde las cepas eran susceptibles o resistentes a un antibiótico [Aan+09].

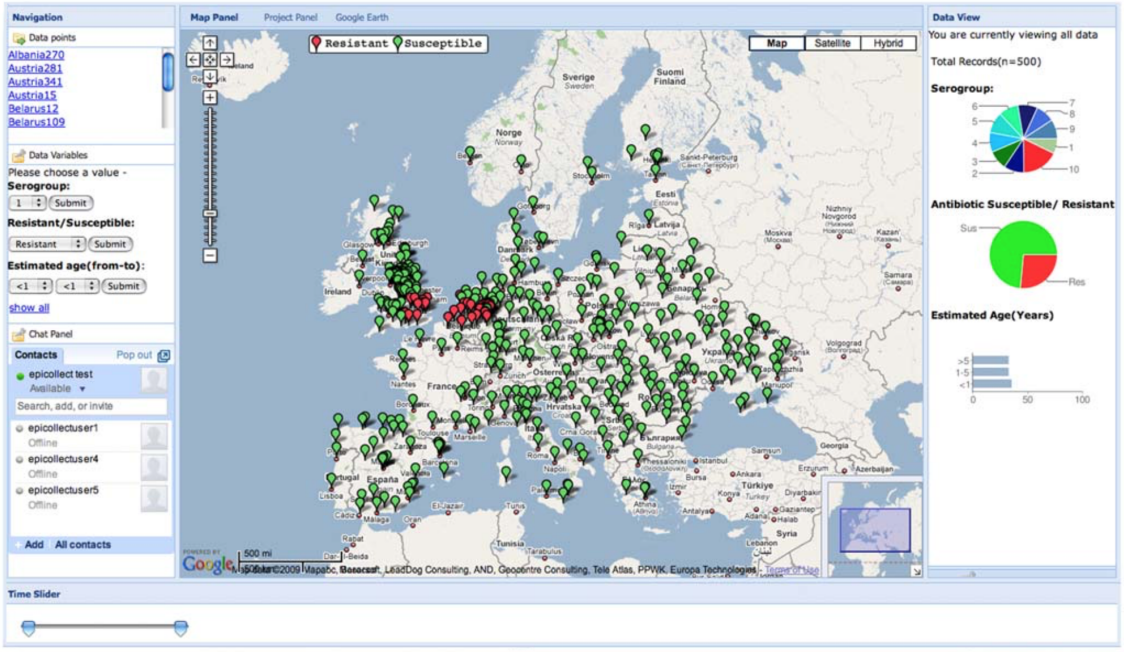


Figura 2.20: Encuesta de animales infectados. Los puntos rojos muestran las cepas resistentes a antibióticos, los verdes los susceptibles

Capítulo 3

Aprendizaje de máquina

En este capítulo se exploran los métodos de aprendizaje de máquina que utilizaremos para realizar predicciones a partir de un conjunto de datos de entrada, además se tratan los métodos de evaluación que nos permitan determinar cuan precisas son las predicciones realizadas. Finalmente se aborda un problema común en el campo del aprendizaje de máquina denominado "clases desbalanceadas". El material de referencia para esta sección incluye principalmente el trabajo de [HPK11] y [ASR15], enriquecidos con otras fuentes que se mencionan a lo largo del capítulo.

3.1. Clasificación

La clasificación de datos es un proceso de dos pasos, que consiste en el aprendizaje (donde se construye el modelo de clasificación o clasificador) y en la clasificación (donde el modelo es usado para predecir la categoría o etiqueta de clase).

En el primer paso, se construye el clasificador describiendo un conjunto clases o conceptos. Este es el paso de aprendizaje (o fase de entrenamiento), en donde el algoritmo de clasificación construye el clasificador analizando o "aprendiendo desde" un conjunto de tuplas con sus correspondientes etiquetas de clases.

Una tupla, X, se representa mediante un vector n-dimensional de atributos, $X = (x_1, x_2, \ldots, x_n)$, representando n medidas hechas en la tupla desde n atributos de la base de datos, respectivamente, $A_1, A_2, \ldots, A_n$. Dado que cada atributo representa una "característica" de X, también se lo denomina como vector de características. Cada tupla que conforma el conjunto de entrenamiento se denominan como tuplas de entrenamiento.

Se asume que cada tupla, X, pertenece a una clase predefinida, el cual se determina mediante otro atributo que se denomina etiqueta de clase. La etiqueta de clase es un atributo con valores discretos y sin orden específico; es categórico (o nominal) en el que cada valor sirve como una categoría o clase.

El primer paso de la clasificación puede ser visto como el aprendizaje de una función de mapeo, $f(X)$, que puede predecir la etiqueta de clase y dada una tupla X. Típicamente esta función de mapeo se representa como reglas de clasificación, árboles de decisión o una formulación matemática.

En el segundo paso, el modelo se utiliza para la clasificación. Primero, se estima

la precisión del clasificador. Si se utiliza el mismo conjunto de entrenamiento para medir la precisión del clasificador resultaría en una estimación optimista, debido a que el clasificador tiende a sobreajustar los datos (por ejemplo, durante la fase de aprendizaje se pudieron haber incorporado algunas anomalías particulares de las tuplas de entrenamiento que no están presentes en el conjunto total de datos). Debido a esto, se utiliza un conjunto de prueba, compuesto de tuplas de pruebas asociados con sus respectivas etiquetas de clase. Éstos son independientes de las tuplas de entrenamiento, puesto que no son utilizadas para construir el clasificador.

La precisión de un clasificador dado un conjunto de pruebas es el porcentaje de tuplas de prueba que son correctamente clasificadas por el clasificador. La etiqueta de clase de la tupla de prueba se compara con la etiqueta que el clasificador ha predicho para la misma tupla.

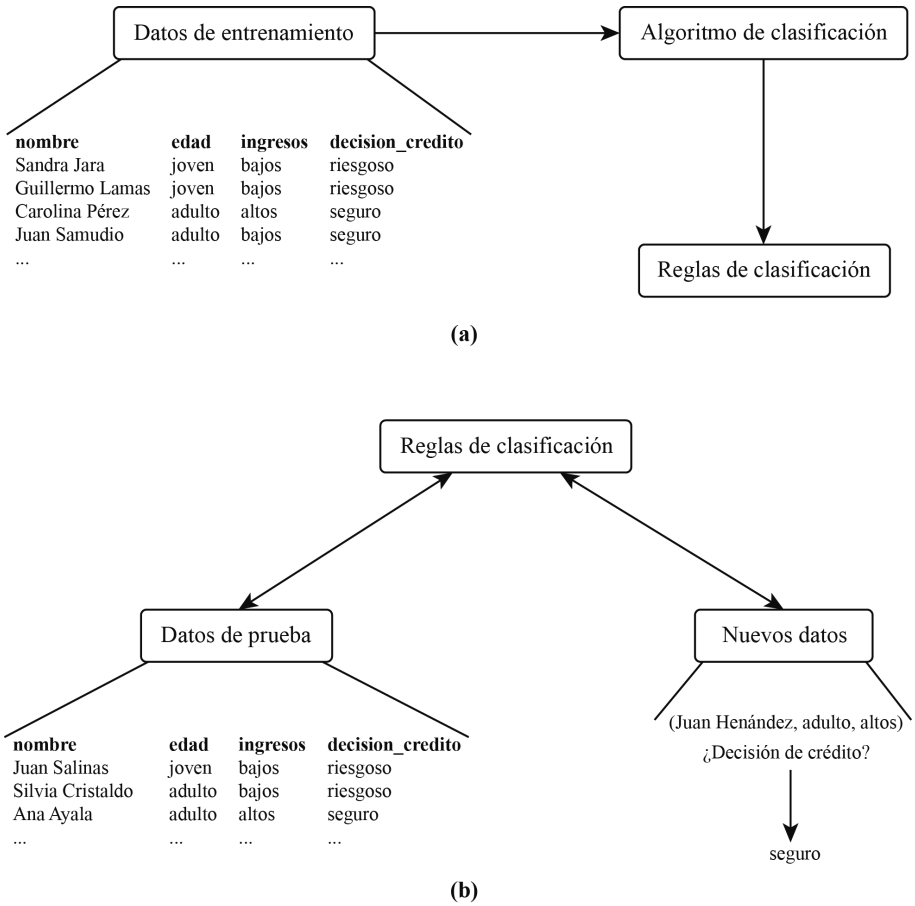


Figura 3.1: El proceso de clasificación de datos

a) Aprendizaje: Los datos de entrenamiento son analizados por el algoritmo de clasificación. La etiqueta de clase representa la decisión de otorgar el crédito

b) Clasificación: Los datos de prueba se utilizan para estimar la precisión del clasificador. Si la precisión se considera aceptable, el clasificador puede ser utilizado para la toma de decisiones.

3.1.1. Árboles de decisión

Un árbol de decisión es una estructura de árbol similar a un diagrama de flujo, donde cada nodo interno (nodo no hoja) denota una prueba en un atributo, cada rama representa un resultado de la prueba, y cada nodo hoja (o nodo terminal) posee una etiqueta de clase. El nodo en la cima del árbol es el nodo raíz. Un árbol de decisión típico se muestra en la Figura 3.2. Representa el concepto *compra_ computadora*, es decir, predice si un cliente en particular compra una computadora. Los nodos internos se indican mediante rectángulos, y nodos hoja se denotan por óvalos. Algunos algoritmos de árboles de decisión sólo producen árboles binarios (donde cada nodo interno se bifurca exactamente a otros dos nodos), mientras que otros pueden producir árboles no binarios.

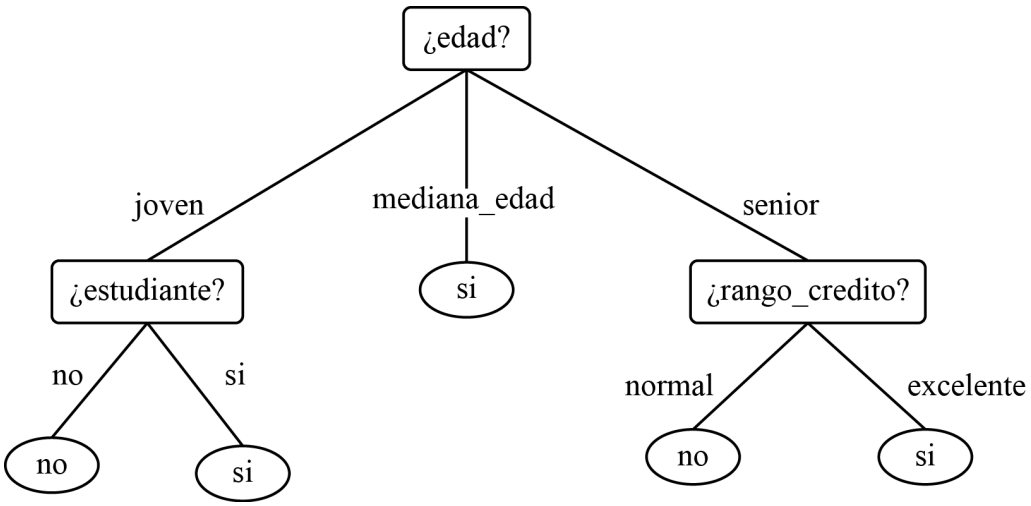


Figura 3.2: Ejemplo de árbol de decisión para el concepto *compra_ computadora*

Dada una tupla, X, para la cual la etiqueta de clase es desconocida, los valores de atributo de la tupla se prueban contra el árbol de decisión. Se traza un camino desde la raíz hasta un nodo hoja, el cual posee la predicción para esa tupla.

Los pasos de aprendizaje y clasificación de los árboles de decisión son sencillos y rápidos. En general, los clasificadores de árboles de decisión tienen una buena precisión, además su representación es intuitiva y generalmente fácil de asimilar para las personas. Los algoritmos de árboles de decisión se han utilizado para la clasificación en muchas áreas de aplicación, tales como la medicina, la fabricación y la producción, análisis financiero, la astronomía y la biología molecular.

Implementación

La mayoría de los algoritmos de clasificación mediante árboles de decisión siguen un enfoque de arriba-abajo[1] , que se inicia con un conjunto de entrenamiento de tuplas y sus etiquetas de clase. El conjunto de entrenamiento es divido recursivamente en subconjuntos más pequeños a medida que el árbol se va construyendo.

[1] También conocido como diseño top-down

El procedimiento básico de construcción de un árbol de decisión se resume en el Algoritmo 1.

Criterio de división

En la línea 8 del algoritmo de generación de árboles de decisión se utiliza la función *Método_ selección_ atributos* para determinar el criterio de división de las tuplas.

El criterio de división nos indica qué atributo probar en el nodo N, además de indicar qué bifurcaciones realizar desde el nodo. Más específicamente, el criterio de división indica el atributo de división y también puede indicar el punto de división o el subconjunto de división. El criterio de división se determina de modo que, idealmente, las particiones resultantes en cada rama sean las más "puras" posibles. Una partición es pura si todas las tuplas pertenecen a la misma clase.

Una *medida para selección de atributos* es una heurística para la selección del criterio de división que "mejor" separa una partición de datos. Proporciona un ranking para cada atributo de las tuplas de entrenamiento, el atributo que posea la mejor puntuación para la medida utilizada es elegido como el atributo de la división de las tuplas.

Si el atributo de la división posee valores continuos o si hay una restricción de utilizar árboles binarios, entonces, respectivamente, o bien un punto de división o un subconjunto de división se determinará también como parte del criterio de división.

Medidas para selección de atributos

Sea D, la partición de datos de entrenamiento con sus respectivas etiquetas de clase. Suponer que la etiqueta de clase tiene m valores distintos definiendo m clases distintas, C_i (para $i = 1, \ldots, m$). Sea $C_{i,D}$ el conjunto de tuplas de la clase C_i en D. Sean $|D|$ y $|C_{i,D}|$ denotan el número de tuplas en D y $C_{i,D}$, respectivamente.

Ganancia de información

La información necesaria para clasificar una tupla en D está dada por

$$Info\,(D) = -\sum_{i=1}^{m} p_i \log_2 (p_i) \tag{3.1}$$

donde p_i es la probabilidad de que una tupla arbitraria en D pertenezca a la clase C_i y se estima mediante ${}^{|C_{i,D}|}/_{|D|}$. $Info\,(D)$ se conoce también como la entropía de D.

Sea

$$Info_A\,(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info\,(D_j) \tag{3.2}$$

la información requerida para clasificar una tupla en D basado en la partición utilizando A.

Algoritmo 1 Generar_árbol_de_decisión: Algoritmo básico para generar árboles de decisión [HPK11]

Entrada:

- Partición de datos, D, que es un conjunto de tuplas de entrenamiento y sus etiquetas de clase asociadas;
- *lista_atributos*, lista de atributos que describe la tupla;
- *Método_selección_atributos*, procedimiento que determina el criterio de partición que "mejor" divide las tuplas de datos en clases individuales. Este criterio consiste de un *atributo_de_división*, y, posiblemente también un *punto_de_división* o un *subconjunto_de_división*

Salida: Un árbol de decisión.
Método:

1) crear un nodo N;
2) **si** las tuplas en D son de la misma clase, C **entonces**
3) **retornar** N como un nodo hoja etiquetada con la clase C;
4) **fin si**
5) **si** *lista_atributos* está vacía **entonces**
6) **retornar** N como un nodo hoja etiquetada con la mayoría de las clases en D; //voto por mayoría
7) **fin si**
8) aplicar **Método_selección_atributos**(D, *lista_atributos*) para buscar el "mejor" *criterio_de_división*;
9) etiquetar el nodo N con el *criterio_de_división*;
10) **si** *atributo_de_división* es discreto **y** se permiten divisiones múltiples **entonces**
11) *lista_atributos* $\leftarrow$ *lista_atributos* $-$ *atributo_de_división*; //remover *atributo_de_división*
12) **fin si**
13) **para cada** división j de *criterio_de_división* **hacer**
14) sea D_j el conjunto de tuplas en D que satisface la división j; //una partición
15) **si** D_j está vacío **entonces**
16) conectar una hoja etiquetada con la clase mayoritaria en D al nodo N;
17) **si no**
18) conectar el nodo retornado por **Generar_árbol_de_decisión**(D_j, *lista_atributos*) al nodo N;
19) **fin si**
20) **fin para**
21) **retornar** N;

La ganancia de información se define como la diferencia entre el requerimiento original de información y el nuevo requerimiento (por ejemplo, después de particionar utilizando A):

$$Ganancia\,(A) = Gain\,(A) = Info\,(D) - Info_A\,(D) \tag{3.3}$$

En otras palabras, $Gain\,(A)$ indica que tanto se gana particionando utilizando A. El atributo A con mayor $Gain\,(A)$ se utiliza como atributo para realizar la partición en el nodo N.

El algoritmo ID3 (Interactive Dichotomiser) utiliza la ganancia de información como método de selección de atributos, desarrollado por Ross Quinlan a finales de los 70. Quinlan luego presentó C4.5 como sucesor de ID3, el cual se convirtió en un modelo con el cual otros algoritmos de clasificación son comparados.

Relación de ganancia

El algoritmo C4.5, sucesor de ID3, utiliza una extensión a la ganancia de información conocida como relación de ganancia (gain ratio). Se aplica un tipo de normalización utilizando un valor denominado "Información de partición" definido análogamente a $Info\,(D)$ como

$$InfoParticion\,(D) = SplitInfo\,(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right) \tag{3.4}$$

Este valor representa el potencial de información generado dividiendo el conjunto de entrenamiento, D, en v particiones, correspondiendo a bifurcaciones del atributo de prueba A. La relación de ganancia se define como

$$RelacionGanancia\,(A) = GainRatio\,(A) = \frac{Gain\,(A)}{SplitInfo_A\,(D)} \tag{3.5}$$

El atributo con la mayor relación de ganancia es utilizado como atributo de partición.

Índice Gini

El índice Gini mide la impureza de D como

$$Gini\,(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{3.6}$$

donde p_i es la probabilidad de que una tupla en D pertenezca a la clase C_i y se estima mediante ${}^{|C_{i,D}|}/_{|D|}$. La suma se computa utilizando m clases.

El índice Gini considera una partición binaria por cada atributo. Considerando la partición binaria, se computa la suma ponderada de la impureza de cada partición.

Por ejemplo, si una partición binaria en A divide D en D_1 y D_2, el índice Gini de D dada la partición es

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \tag{3.7}$$

Para cada atributo, cada una de las posibles particiones binarias es considerada.

La reducción de la impureza que se incurriría por una división binaria en el atributo A es

$$\Delta Gini(A) = Gini(D) - Gini_A(D) \tag{3.8}$$

El atributo que maximiza la reducción de impureza (o, equivalentemente tiene el mínimo índice Gini) se selecciona como atributo para la partición.

3.2. Evaluación y selección de modelos

Una vez que se ha construido el modelo de clasificación, se necesita estimar cuán preciso es el clasificador para predecir la etiqueta de clase. Además, a menudo es útil comparar el desempeño de diferentes clasificadores de manera a determinar qué clasificador funciona mejor en un conjunto de datos [Tan+06].

3.2.1. Métricas de evaluación

Antes de presentar las métricas, se procederá a introducir algunas terminologías. Las **tuplas positivas** constituyen las clases de interés principal, mientras que las **tuplas negativas** son las restantes. Por ejemplo, dadas dos clases, las tuplas positivas pueden ser *compra_computadora = si* mientras que las tuplas negativas constituyen las que poseen *compra_computadora = no*.

Suponer que se tiene un modelo de clasificación con tuplas de prueba con sus respectivas etiquetas de clase. Sean P el número de tuplas positivas y N el número de tuplas negativas. Para cada tupla, se compara la predicción del clasificador con la etiqueta de clase conocida. Se introducen además los siguientes términos

- **Verdaderos positivos** (VP): Se refiere a las tuplas positivas que fueron correctamente etiquetadas por el clasificador. Sea VP el total de verdaderos positivos.
- **Verdaderos negativos** (VN): Tuplas negativas que fueron correctamente etiquetados por el clasificador. Sea VN el número de verdaderos negativos.
- **Falsos positivos** (FP): Son tuplas negativas que fueron incorrectamente etiquetadas como positivas. Sea FP el número de falsos positivos.
- **Falsos negativos** (FN): Son tuplas positivas que fueron mal etiquetadas como negativas. Sea FN el número de falsos negativos.

		Clase predicha		
		si	no	Total
Clase real	si	VP	FN	P
	no	FP	VN	N
	Total	P'	N'	$P+N$

Figura 3.3: Matriz de confusión

Éstos términos se resumen en la matriz de confusión de la Figura 3.3.

La matriz de confusión es una herramienta útil para analizar que tan bien el clasificador puede reconocer tuplas de diferentes clases. Dadas m clases (donde $m \geq 2$), una matriz de confusión es una tabla de al menos m por m. Una entrada, $CM_{i,j}$ en las primeras m filas y m columnas indican el número de tuplas de clase i que fueron etiquetadas por el clasificador como clase j. En un clasificador con buena exactitud, la mayoría de las tuplas estarían representadas a lo largo de la diagonal de la matriz de confusión, desde la celda $CM_{1,1}$ hasta $CM_{m,m}$, con el resto de las celdas siendo cero o cercano a cero, es decir, FP y FN poseen un valor cercano a cero.

La tabla puede tener filas y columnas adicionales de manera a mostrar totales. Por ejemplo, en la matriz de confusión de la Figura 3.3, se muestran P y N. Además, P' es el número de tuplas que fueron etiquetadas como positivas ($VP+FP$) y N' es el número de tuplas que fueron etiquetadas como negativas ($VN+FN$). El número total de tuplas es $VP+VN+FP+FN$, o $P+N$, o $P'+N'$. Se debe tener en cuenta que la matriz de confusión que se muestra sirve para un problema de clasificación binaria, sin embargo, las matrices de confusión pueden ser extendidas fácilmente a problemas con múltiples clases. A continuación se introducen algunas métricas de evaluación.

Exactitud Dado un conjunto de entrenamiento, la exactitud (accuracy) de un clasificador es el porcentaje de tuplas de entrenamiento que fueron correctamente clasificadas por el clasificador.

$$exactitud = \frac{VP + VN}{P + N} \tag{3.9}$$

La exactitud también se conoce como la *tasa de reconocimiento* del clasificador, es decir, refleja que tan bien el clasificador reconoce tuplas de varias clases.

Tasa de error También conocido como tasa de errores (error rate) de clasificación, el cual es simplemente $1 - exactitud(M)$, donde $exactitud(M)$ es la exactitud del clasificador M, también puede ser calculado como

$$tasa\ \ de\ \ error = \frac{FP + FN}{P + N} \tag{3.10}$$

Si se utiliza el conjunto de entrenamiento en vez del conjunto de pruebas para estimar el error de un modelo entonces esta métrica recibe el nombre de *error de*

resubstitución. En este caso la estimación del error es optimista, debido a que no se ha probado en ejemplares de tuplas que aún no han sido vistas.

Sensibilidad y especificidad La sensibilidad (sensitivity) se conoce también como la tasa (de reconocimiento) de verdaderos positivos (la proporción de tuplas positivas que son correctamente clasificadas), mientras que la especificidad (specificity) es la tasa de verdaderos negativos (la proporción de tuplas negativas que son correctamente identificadas). Estas métricas se definen como

$$sensibilidad = \frac{VP}{P} \tag{3.11}$$

$$especificidad = \frac{VN}{N} \tag{3.12}$$

Precisión y exhaustividad La precisión (precision) se define como la tasa de tuplas etiquetadas como positivas que actualmente lo son, mientras que exhaustividad es la tasa de tuplas positivas que fueron etiquetadas como tal. La exhaustividad (recall) es igual a la sensibilidad (tasa de verdaderos positivos), estas métricas pueden ser calculadas como

$$precisión = \frac{VP}{VP + FP} \tag{3.13}$$

$$exhaustividad = \frac{VP}{VP + FN} = \frac{VP}{P} \tag{3.14}$$

Una precisión perfecta de 1,0 para una clase C significa que cada tupla que el clasificador ha etiquetado que pertenece a la clase C en realidad pertenece a la clase C. Sin embargo, no dice nada acerca del número de tuplas con clase C que el clasificador ha errado. Un valor de exhaustividad perfecto de 1,0 para C significa que cada ítem de la clase C fue etiquetado como tal, pero no dice cuántas otras tuplas fueron incorrectamente etiquetadas como pertenecientes a la clase C.

Típicamente los valores de precisión y exhaustividad son usados en conjunto. Una forma de combinar dichos valores en una sola métrica se denominan como *valor-F* (conocido además como *valor* F_1 o *medida-F*) y *valor-*F_β.

El *valor-F* se considera como una media armónica que combina los valores de la precisión y de la exhaustividad. De tal forma que,

$$F = \frac{2 \times precisión \times exhaustividad}{precisión + exhaustividad} \tag{3.15}$$

La fórmula general para un número real β es,

$$F_\beta = \frac{(1 + \beta^2) \times precisión \times exhaustividad}{\beta^2 \times precisión + exhaustividad} \tag{3.16}$$

Si β es igual a uno, se está dando la misma ponderación (o importancia) a precisión que a la exhaustividad, si β es mayor que uno de damos más importancia a exhaustividad, mientras que si es menor que uno se le da más importancia a la precisión [Bei06]. Las métricas más comúnmente utilizadas para F_β son F_2 y $F_{0,5}$.

Método de retención

Con el método de retención los datos son particionados aleatoriamente en dos conjuntos independientes, un conjunto de entrenamiento y un conjunto de prueba. Típicamente, dos tercios de los datos se utilizan para entrenamiento (para construir el modelo de clasificación) y el tercio restante se coloca en el conjunto de pruebas. La exactitud del modelo se estima con el conjunto de prueba (Figura 3.4). La estimación es pesimista debido a que solo una porción de los datos iniciales son utilizados para entrenar el modelo.

En el *remuestreo aleatorio* el método de retención se repite k veces. La exactitud global se estima mediante el promedio de las exactitudes obtenidas en cada iteración.

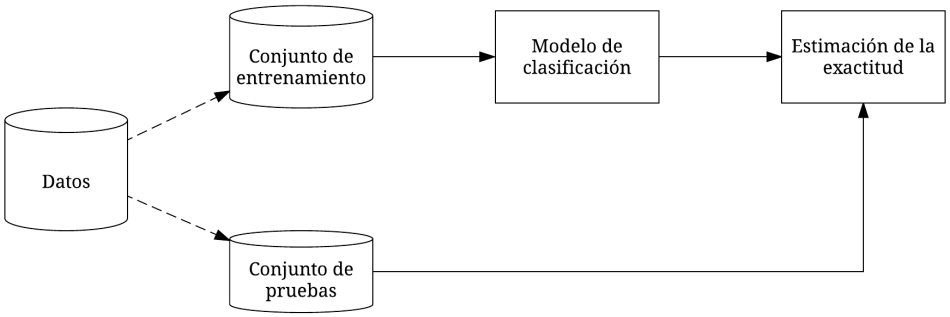


Figura 3.4: Estimación de la exactitud utilizando el método de retención

Validación cruzada

En la validación cruzada de k iteraciones [2] , los datos iniciales son particionados aleatoriamente en k subconjuntos mutuamente excluyentes, $D_1, D_2, \ldots, D_k$, aproximadamente del mismo tamaño. El entrenamiento y prueba se realiza k veces. En la iteración i, la partición D_i se reserva como conjunto de prueba, y las particiones restantes se utilizan para entrenar el modelo. A diferencia del método de retención y remuestreo aleatorio, cada subconjunto se utiliza el mismo número de veces para el entrenamiento y una vez para pruebas. La estimación de la exactitud es el número total de clasificaciones realizadas correctamente desde las k iteraciones, divido el número total de tuplas en los datos iniciales.

En la validación cruzada estratificada, los subconjuntos son estratificados de tal manera a que la distribución de las clases de las tuplas en cada subconjunto sea aproximadamente igual a los datos iniciales.

En general, se recomienda utilizar la validación cruzada estratificada de 10 iteraciones para la estimación de la exactitud debido a su relativo bajo sesgo y varianza [HPK11; WF05].

[2] Conocido también como k-fold cross-validation

3.2.2. Selección de modelos

Curvas ROC

Las curvas ROC (Receiver Operating Characteristic) son una herramienta visual que permite la comparación de dos modelos de clasificación. Una curva ROC para un modelo en particular muestra el compromiso entre la tasa de verdaderos positivos (TVP), y la tasa de falsos positivos (TFP). Dado un modelo de clasificación y un conjunto de pruebas, TVP es la proporción de tuplas positivas que fueron correctamente etiquetadas por el clasificador; TFP es la proporción de tuplas negativas que fueron erróneamente etiquetadas como positivas. Dados VP, FP, P, y N, descriptos en la sección 3.2.1, se tiene que $TVP = {}^{VP}/_{P}$, el cual equivale a la *sensibilidad* y $TFP = {}^{FP}/_{N}$ que es igual a $1 - especificidad$.

Para un problema de dos clases, la curva ROC permite visualizar el compromiso entre la tasa en la que el modelo puede reconocer con exactitud los casos positivos versus la tasa en la que erróneamente identifica casos negativos como positivos para diferentes porciones del conjunto de prueba. Un incremento en el valor de TVP ocurre con el costo de un incremento del valor de TFP.

Área bajo la curva (AUC) Para estimar la exactitud del modelo se puede calcular el área bajo la curva ROC. Esta área posee un valor comprendido entre 0,5 y 1, donde 1 representa un valor predictivo perfecto y 0,5 una prueba sin capacidad discriminatoria. Por esto, siempre lo deseable es que esta medida sea la más alta posible [Jar14].

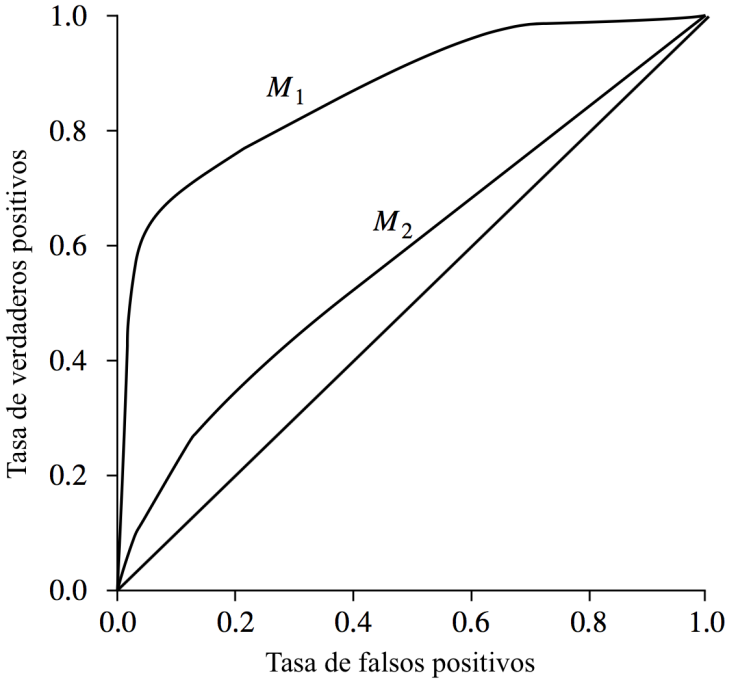


Figura 3.5: Curvas ROC de dos modelos de clasificación, M_1 y M_2. La diagonal muestra que por cada verdadero positivo, es igualmente posible encontrar un falso positivo. Mientras más cerca está la curva ROC de la diagonal, menos exacto es el modelo. En este caso, el modelo M_1 es más exacto.

3.3. Clases desbalanceadas

Se denomina problema de clases desbalanceadas, o problema de clases no balanceadas cuando la clase de interés principal está representada por unas pocas tuplas. Esto es, la distribución del conjunto de datos refleja una significante mayoría de las clases negativas y una minoría de las clases positivas [HPK11]. El grado de desbalanceo de una distribución puede ser denotado por la proporción del tamaño de la muestra de la clase minoritaria a la de la clase prevalente. En aplicaciones prácticas, la proporción puede ser tan drástica como 1:100, 1:1000, o superior [SWK09]. En los diagnósticos médicos, por ejemplo, es mucho más costoso diagnosticar falsamente a un paciente cancerígeno como sano (un falso negativo) que diagnosticarlo erróneamente con cáncer (un falso positivo). Un error de falso negativo puede llevar a la pérdida de una vida y debido a eso es mucho más costoso que un error de falso positivo. Otras aplicaciones que implican datos con clases desbalanceadas incluyen, detección de fraude, detección de derrames de petróleo a partir de imágenes satelitales y el monitoreo de fallas [HPK11].

Los algoritmos de clasificación tradicionales apuntan a minimizar el número de errores cometidos durante la clasificación. Ellos asumen que el costo del falso positivo y el falso negativo son iguales. Por lo tanto, el hecho de asumir una distribución balanceada de clases e igual costo de errores, no son adecuados para datos con clases no balanceadas [HPK11].

Considérese un escenario con un grado de desbalanceo de 1:100. La proporción sugiere que por cada tupla de la clase minoritaria (positiva) existen 100 tuplas mayoritarias (negativas). Un algoritmo de clasificación que tiene como objetivo maximizar la exactitud, producirá una exactitud del 99 % clasificando correctamente todas las tuplas de la clase mayoritaria pero clasificando erróneamente una tupla de la clase minoritaria [ASR15]. En estas situaciones, es una característica de las distribuciones desbalanceadas que los clasificadores estén sesgados hacia la clase mayoritaria y muestren un pobre rendimiento hacia las clases minoritarias, debido a que clasifican todo como clase mayoritaria ignorando por completo a la clase minoritaria [LD13].

3.3.1. Enfoques a las clases desbalanceadas

La literatura sugiere varios algoritmos y técnicas para resolver el problema de la distribución desbalanceada de los datos. Los enfoques se dividen principalmente en tres métodos tales como muestreo, algoritmos y selección de atributos [LD13].

Muestreo

Las técnicas de muestreo se utilizan para resolver el problema de la distribución el conjunto de datos, estas técnicas envuelven un remuestreo artificial de los datos, también conocido como método de preprocesamiento de datos. El muestreo se puede realizar de dos maneras, mediante el submuestreo de la clase mayoritaria, sobremuestreo de la clase minoritaria o la combinación de ambas [LD13].

Submuestreo El método más importante del submuestreo es el submuestreo aleatorio, en el cual se trata de balancear la distribución de las clases eliminando la clase mayoritaria. La figura 3.6 muestra el método del submuestreo aleatorio. El problema con este método es la pérdida de información importante.

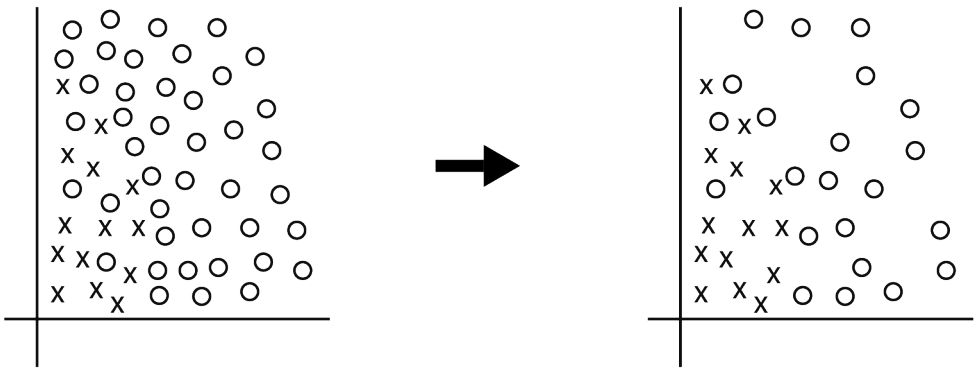


Figura 3.6: Submuestreo: Se eliminan ejemplares de la clase mayoritaria

Sobremuestreo Métodos de sobremuestreo aleatorio también ayudan a alcanzar una distribución equitativa de las clases replicando los ejemplares de la clase minoritaria.

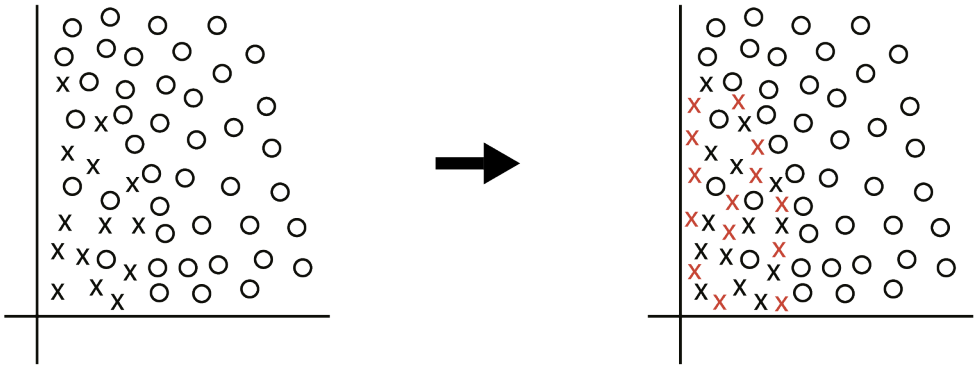


Figura 3.7: Sobremuestreo: Se replican ejemplares de la clase minoritaria

Existen muchas variaciones de las técnicas de submuestreo y sobremuestreo, pueden variar, por ejemplo en como las tuplas son añadidas o eliminadas.

SMOTE *Synthetic Minority Over-Sampling Technique* es un algoritmo basado en muestreo que fue introducido para tratar solventar el problema del desbalanceo de clases. Es una de las principales técnicas dada su simplicidad y efectividad en donde sobremuestreo no se realiza replicando clases minoritarias sino que construyendo nuevas instancias sintéticas en el segmento de que une una tupla minoritaria con uno de sus K vecinos más cercanos [Cha+02].

Las instancias sintéticas se generan de la siguiente manera:

Algoritmo 2 Algoritmo SMOTE [Chi13]

1) **para cada** instancia de la clase minoritaria C **hacer**
2) $vecinos \leftarrow$ obtener $\boldsymbol{KNN}(5)$
3) $N \leftarrow$ seleccionar aleatoriamente uno de los *vecinos*
4) crear una nueva instancia de la clase minoritaria R usando los atributos de C y el vector resultado de la diferencia de N y C multiplicado por un número aleatorio:
   $R.atributos \leftarrow C.atributos + (C.atributos - N.atributos) \times \boldsymbol{random}(0,1)$
5) **fin para**

La intuición detrás de la construcción del algoritmo es que el sobremuestreo produce sobreajuste debido a que las instancias repetidas causa que los límites de la decisión se estrechen. En vez de eso, se crean ejemplares similares. Para el clasificador

estas nuevas instancias no son copias exactas que por consecuencia mediante los cuales se consigue mejorar el espacio de decisión de los clasificadores [Cha+02].

Algoritmos

Varios algoritmos fueron creados para resolver el problema de las clases desbalanceadas. El objetivo de este enfoque es optimizar el desempeño de los algoritmos de clasificación.

Métodos de aprendizaje de una sola clase reconocen tuplas que pertenecen a esa clase y rechazan las demás. Bajo ciertas condiciones, como conjuntos de datos multidimensionales, los algoritmos de aprendizaje de una sola clase dan mejores resultados que otros.

En lugar de cambiar la distribución de las clases, aplicar costos en la toma de decisiones es otra manera de mejorar el desempeño del clasificador. Los métodos de aprendizaje sensibles al costo tratan de maximizar una función de pérdida asociada con un conjunto de datos. Estos métodos de aprendizaje están motivados por el hecho de que la mayoría de las aplicaciones del mundo real no tienen costos uniformes para las clasificaciones erróneas. Los costos reales asociados con cada tipo de error se desconocen normalmente, por lo que estos métodos deben determinar la matriz de costos basada en los datos y aplicarlo a la etapa de aprendizaje [LD13].

Selección de atributos

El objetivo de la selección de atributos, en general, es seleccionar el subconjunto de atributos que permite al clasificador alcanzar un óptimo desempeño.

Para conjuntos de datos de muchas dimensiones, utiliza filtros que califican cada atributo independientemente en base a una regla. La selección de atributos representa un paso clave para muchos algoritmos de aprendizaje de máquina, especialmente cuando los datos poseen alta dimensionalidad [LD13].

La selección de atributos se utiliza en clasificación con clases desabalanceadas principalmente para definir la relevancia de los atributos para la clase objetivo, es decir se utiliza para medir la bondad de un atributo. La selección de atributos ayuda a sugerir atributos con mucha influencia los cuales a menudo proveen información intrínseca y discriminante entre las clases [ASR15].

Capítulo 4

Implementación

El desarrollo de este proyecto tiene como objetivo agilizar el proceso de recolección de datos mediante el uso de dispositivos móviles, los cuales una vez conectados a internet permitirán la disponibilización casi inmediata de los datos recabados, permitiendo el análisis de los mismos de modo a facilitar la toma de decisiones. Este capítulo se organiza de la siguiente manera: en la primera sección se describe el modelo propuesto y arquitectura diseñada para la representación de ese modelo. En la segunda sección se describen las herramientas y procedimientos realizados para la implementación del modelo propuesto aplicado a un caso de estudio.

4.1. Modelo Propuesto

4.1.1. Descripción General

Se plantea la utilización de un sistema de recolección de datos que opere en teléfonos móviles y tablets que facilite el proceso de recolección de datos, comprendiendo todas sus etapas: construcción, recolección y visualización.

1. Construcción: consiste en la definición y creación del formulario electrónico a ser utilizado.

2. Recolección: comprende la etapa en donde un encuestador obtiene los datos y los da de alta utilizando la interfaz móvil del formulario definido en el paso 1.

3. Visualización: consiste en la visualización remota de los datos recogidos por los encuestadores.

Posteriormente al proceso de recolección de datos en todas sus etapas, se plantea la utilización de la información recogida para someterla a un análisis que permitirá la predicción de una variable que fue recolectada durante la fase de recolección. Esta herramienta permitirá la extracción de conocimiento, y una mejor comprensión de fenómenos asociados a un tema de interés en particular.

Recolección de datos

La recolección de datos cuenta con dos módulos principales: cliente móvil y servidor.

El cliente móvil es el software capaz de interpretar la definición del formulario, desplegándolo en pantalla de manera a que el encuestador pueda utilizarlo para recoger los datos. Debe reunir las siguientes características:

- Capacidad de ejecutarse en teléfonos móviles o tablets de bajo coste.
- Capacidad de presentarse en formato web para visualización en computadoras de escritorio y laptops.
- Recolección de datos de texto, selección simple y múltiple.
- Recolección multimedia: imágenes, audio, video.
- Recolección de datos de geolocalización.
- Almacenamiento sin conexión de los formularios completos.
- Capacidad de sincronización online de formularios completos.

El módulo servidor consiste en un servicio online en internet y una interfaz gráfica de usuario que debe reunir las siguientes características:

- Disponibilización de definiciones de formularios.
- Creación de un repositorio para los formularios enviados.
- Visualización en tiempo real de formularios completos enviados.
- Visualización de los formularios georeferenciados en un mapa.

Ésto permitirá el despliegue de encuestas de campo, recogiendo datos enriquecidos sin necesidad de contar con una conexión activa a internet. Se persigue además, que sea de fácil utilización para personas no técnicas, consiguiendo de ésta manera la independencia para llevar a cabo el proceso completo de recolección y extracción de datos que servirán para la toma de decisiones.

Es fundamental mencionar, además, que los módulos cliente móvil y servidor deben ser de código abierto lo que implicará no incurrir en costos a la hora de montar la infraestructura que soportará el proceso completo de recolección y análisis. El hecho de contar con software de código abierto abre la posibilidad de adecuar los módulos de acuerdo a necesidades específicas de acuerdo a cada caso.

Análisis de datos

De manera a poder brindar soporte para una mejor comprensión de los datos y una consecuente toma de decisiones más acertada se propone la utilización de algoritmos de aprendizaje de máquina de manera a realizar tareas de clasificación. Mediante la creación de modelos de clasificación se espera poder visualizar como se relacionan los datos, de manera a poder extraer reglas que permitan graficar un problema en particular.

La clasificación, como se vió en la sección 3.1, consiste en el aprendizaje desde un conjunto de datos utilizado para predecir un valor de salida, también denominado clase. Por ejemplo, si se realiza un proceso de recolección de datos utilizando un formulario en donde se recogen datos sobre el nivel de infestación de mosquitos. Suponiendo que se recoge información de las características del entorno de la casa y un indicador de infestación de mosquitos en la vivienda. Una vez recogidos los datos, se aplica un modelo de clasificación que permitirá tener las características de los entornos de las viviendas que son más susceptibles a ser infestados por mosquitos.

4.2. Implementación

4.2.1. Recolección de datos

En el Capítulo 2 se describieron las herramientas de recolección de datos de código abierto disponibles, analizando y comparando dichas herramientas (ver tabla 4.1).

Se concluye que la suite Open Data Kit (ODK) es la herramienta que mejor se ajusta a las características requeridas por el modelo propuesto. Se tuvieron en cuenta los siguientes factores para la elección de la herramienta:

Módulo cliente

- **Plataforma:** Los sistemas operativos móviles más utilizados y difundidos en el mercado actual son Android e iOS, siendo Android la más difundida a nivel mundial con 86.2 % de cuota del mercado [Gar]. Con estos datos se puede concluir que la plataforma a soportar deberá ser Android debido a su gran difusión, lo que se traduce en más dispositivos disponibles para utilizar la herramienta. ODK dispone del módulo ODK Collect, que está diseñada exclusivamente para dispositivos que utilizan el sistema operativo Android, con soporte desde la versión Gingerbread (2.3.3) para arriba, lo que significa una disponibilidad de hasta el 99,9 % de dispositivos móviles con Android limitado por las características del dispositivo como ser almacenamiento, conectividad o sensores GPS. [Dev].

- **Coste:** Los dispositivos con sistema operativo Android son manufacturados por varios fabricantes a nivel mundial, lo cual disponibiliza una amplia gama de dispositivos de diferentes características y precios. Existen dispositivos Android de coste bastante accesible con características suficientes para que

cualquier herramienta de recolección de datos funcione correctamente ayudando de esta forma a reducir el presupuesto destinado a compra de dispositivos. Las características mínimas de los dispositivos necesarios para que ODK Collect se ejecute sin limitar sus funcionalidades se describen a continuación:

- Sistema Operativo Android (mínima versión 4.1 - Jellybean)
- Acceso a Internet (a través de WiFi o redes móviles)
- Sensor GPS.
- Cámara (recomendado 5Mpx para arriba para mejor definición de las fotos) y micrófono.
- Soporte para tarjetas MicroSD.

- **Características técnicas:** Se valora el soporte a distintos tipos de datos, como también validaciones de datos y controles lógicos del flujo de la toma de datos. ODK Collect soporta una gran variedad de tipos de datos en preguntas, además de validaciones avanzadas de datos, permitir bifurcaciones y repeticiones de preguntas, lo cual lo convierte en una herramienta completa a la hora de diseñar formularios.

- **Manejo de datos:** El soporte de almacenamiento de datos offline y la sincronización de datos a través de internet son características requeridas para toma de datos previendo casos como toma de datos en lugares donde no exista conexión a internet o la conexión de internet sea inestable. ODK Collect permite el almacenamiento de datos en la tarjeta SD del dispositivo para su posterior sincronización a través de internet cuando el usuario lo confirme.

Módulo servidor

- **Diseñador de formularios:** Se necesita una herramienta capaz de generar definiciones de formularios a través de una interfaz gráfica amigable para usuarios no técnicos. ODK dispone de la herramienta ODK Build en donde permite a los creadores de formularios usar una interfaz de tipo "arrastrar y soltar" para la creación interactiva de formularios.

- **Repositorio de datos:** El almacenamiento y manejo de los datos recolectados necesita ser manejado por el servidor para su posterior accesibilidad. ODK Aggregate dispone de una interfaz para extracciones de datos en formatos populares como csv e integraciones a sistemas existentes usando HTTP, además de estar diseñado para ser un almacén genérico de datos compatible con cualquier plataforma que soporte Java.

- **Acceso a datos en tiempo real:** El acceso a datos en tiempo real posibilita un análisis de datos cuasi inmediato y por ende toma de decisiones más precisas en casos de encuestas sobre crisis endémicas. ODK Aggregate provee una interfaz web para visualizar los datos de una o más encuestas en formato de tablas. Los datos de disponibilizan según se van cargando a la base de datos.

- **Visualización de datos georeferenciados:** La visualización de datos georefenciados en un mapa permite un mejor entendimiento y posterior análisis de los datos recabados. ODK Aggregate ofrece una vista de mapas en donde se muestran todos los datos recolectados con georeferencias disponibles.

Comparativa de Herramientas Analizadas			
Características	**Open Data Kit**	**Epi Info**	**EpiCollect**
	Cliente		
Plataformas Soportadas	✓	✓	✓
Coste	✓	✓	✓
Características Técnicas	✓	✓	✗
Manejo de Datos	✓	✓	✓
	Servidor		
Diseñador de Formulario	✓	✓	✓
Repositorio de Datos	✓	✗	✗
Acceso a Datos en Tiempo Real	✓	✓	✓
Visualización de Datos Georefenciados	✓	✓	✓

Tabla 4.1: Comparativa de Herramientas Analizadas. Las celdas sombreadas indican que la herramienta cumple parcialmente con la característica

4.2.2. Configuración de ODK

Para poder utilizar ODK se necesita seguir los siguientes pasos:

- Diseñar un formulario.
- Configurar un servidor para ejecutar ODK Agreggate.
- Instalar ODK Collect en un dispositivo compatible y conectado a la red para enviar datos al servidor.

A continuación se describe cada uno de estos pasos.

Diseño de formulario

Para el diseño de formularios se pueden utilizar alguna de estas dos herramientas o la combinación de ambas:

- ODK Build
- XLSForm

ODK Build permite realizar el diseño de formularios a través de una interfaz gráfica web, gracias a la técnica de "Arrastrar y soltar", sin embargo, esta herramienta está limitada para formularios básicos, ya que ante la aparición de complejidades como grupos de repetición de preguntas y preguntas condicionales se vuelve más complicado de utilizar al no permitir la edición de los datos generados con total libertad. En caso de necesidad de diseño de formularios complejos, se recomienda utilizar XLSForm, que es un estándar creado para simplificar la creación de formularios en Excel. La creación se realiza en un formato legible para humanos usando la herramienta familiar que la mayoría conoce - Excel.

Configuración de servidor con ODK Aggregate

La instalación de ODK Aggregate se puede hacer desplegando el servidor y repositorio de datos provisto en dos formas:

- Desplegar en Google App Engine, permitiendo a los usuarios correr el servicio rápidamente sin tener que enfrentar las complejidades de configurar un servicio web escalable.
- Desplegar en un servidor Tomcat (o cualquier contenedor web compatible con web servlets versión 2.5 para arriba) con un servidor de bases de datos MySQL o PostgreSQL.

A continuación se detallan los pasos para desplegar ODK Aggregate en un servidor Tomcat:

- Definir los requerimientos del servidor e instalar el servidor (este paso se puede omitir si ya se cuenta con un servidor configurado). Estos requerimientos dependerán del uso que se dará al servidor que se puede determinar gracias a los siguientes criterios:
    - **Disponibilidad:** Si se necesita que el servidor esté siempre disponible o no. Dependerá de qué tan crítico es tener los datos en tiempo real. La mayoría de los casos no necesitan un servidor con alta disponibilidad dada que la frecuencia con la que se sincronizan los datos es muy baja durante el periodo de recolección de datos de un estudio.
    - **Pérdida de datos:** Depende de si se puede tolerar o no pérdida de datos debidos a posibles fallos. Se recomienda invertir en métodos de respaldo diarios si no se puede permitir pérdida de datos de menos de 24 horas.
    - **Tamaño de conjunto de datos:** La cantidad de datos que se estima recolectar impacta proporcionalmente sobre el tamaño de disco de la máquina que albergue ODK Agreggate y el servidor de base de datos.
    - **Seguridad y protección de datos:** Si se necesita prevenir escuchas o visualización de datos privados mientras se envían al servidor con ODK Agreggate es recomendable conectar ODK Agreggate sólo dentro de la red de la organización ó obtener un certificado SSL e instalarlo en el servidor Tomcat de modo a encriptar y asegurar los datos transmitidos.

- Instalar Tomcat en el servidor. Los pasos generales son:
  - Instalar Java 7 o superior.
  - Configurar la variable PATH con el directorio donde se encuentra instalado Java.
  - Descargar e instalar Tomcat 8.
- Configurar el servidor y dispositivos de red para que acepten conexiones externas de otros dispositivos (por ejemplo dispositivos Android). Esto comprende los siguientes pasos:
  - Configurar el cortafuegos del servidor.
  - Hace visible el servidor desde internet (en caso de ser necesario).
  - Establecer un nombre DNS para el servidor.
- Si se necesita acceso seguro (https), obtener e instalar un certificado SSL.
- Elegir e instalar un servidor de base de datos (puede ser MySQL o PostgreSQL o Microsoft SQL Server o Azure SQL).
- Descargar e instalar ODK Agreggate. El instalador contiene una guía para configurar ODK Aggregate para Tomcat y el servidor de base de datos elegido. Luego genera un archivo WAR (web archive) que contiene el servidor ODK Aggregate configurado, un script `create_db_and_user.sql` para crear la base de datos y usuario que ODK Aggregate utilizará para acceder a la misma y también un archivo Readme.html con instrucciones para finalizar la instalación. Además se requerirá completar datos como el nombre de host del servidor ODK Aggregate (aquí se completa con el nombre DNS previamente establecido), el nombre de la base de datos, el usuario y contraseña de los mismos. Una vez terminada la configuración, se puede acceder a la aplicación a través de la URL configurada. Al principio, ODK Aggregate aceptará envíos anónimos de datos de ODK Collect y accesos no autenticados para visualizar los datos de administración de formularios. Si la pestaña "Site admin" no está visible, entonces se debe ingresar a la aplicación con el super usuario creado durante la instalación de ODK Aggregate. De esta forma se habilitará la pestaña "Permissions" donde se puede crear usuarios de ODK Aggregate y otorgar permisos a los mismos. Los permisos disponibles son:
  - *Data Collector*: permisos para consultar datos de formulario y enviar datos recolectados.
  - *Data Viewer*: visualizar datos recolectados desde la aplicación web de ODK Aggregate.
  - *Form Management*: administrar la subida de formularios y exportar datos.
  - *Site Admin*: administra los usuarios y datos en general de la aplicación.

Una vez creados los usuarios sólo resta que uno de ellos con permisos necesarios para subir los datos de formularios, suba el archivo creado con ODK Build o con XLSForm para disponibilizar el uso del mismo para los usuarios de ODK Collect.

Instalación de ODK Collect en dispositivos

La instalación de ODK Collect en dispositivos Android compatibles se puede realizar de dos formas:

- Descarga desde el Play Store
- Descarga desde la web

A continuación se detallan los pasos a seguir para ambas formas de instalación:

Descarga desde el Play Store

- Seleccionar la aplicación "Play Store" desde el menú de aplicaciones del teléfono.
- Al iniciar la aplicación, buscar "ODK" en la misma y buscar la aplicación "ODK Collect" del desarrollador "Open Data Kit".
- Seleccionar la aplicación e tocar el botón de instalar. Aceptar los permisos requeridos para instalarlo.

Descarga desde la Web

- Seleccionar la aplicación "Ajustes" o "Configuración" desde el menú de aplicaciones del teléfono.
- Seleccionar la opción "Aplicaciones" o "Seguridad". Buscar la opción "Orígenes desconocidos" y asegurarse de que esté marcada/activada.
- Volver al menú de aplicaciones y seleccionar el navegador.
- Navegar a *https://opendatakit.org/downloads/download-category/collect/* y descargar la última versión de ODK Collect.
- Al descargar todo, seleccionar el archivo descargado desde la aplicación de "Descargas" y aprobar todas las configuraciones de seguridad para instalarlo.

Una vez descargada e instalada la aplicación, esta aparecerá en el menú de aplicaciones. Para iniciar, seleccionar el ícono de la aplicación.

Configuración La aplicación debe ser configurada de forma a que pueda consultar datos de formularios desde un servidor con ODK Aggregate y enviar los datos recabados al servidor. Estos son los pasos para configurar la aplicación:

- En la pantalla principal, tocar en el botón de menú del costado superior derecho y seleccionar "Cambiar Configuración".
- Entre las opciones disponibles, seleccionar "Servidor".
- Editar los campos "URL", "Nombre de usuario" y "Contraseña" con los datos de la URL del servidor con ODK Aggregate a utilizar y el nombre de usuario y contraseña si fuese requerido por la configuración del servidor.

4.2.3. Análisis de datos

Para el análisis de los datos se implementa una API REST, en donde se pueden acceder a los formularios disponibles registrados en el servidor ODK Aggregate, visualizar sus datos y realizar operaciones de análisis. La tabla 4.2 muestra las principales operaciones disponibles en la API.

Operación	Descripción
`GET /forms`	Obtiene la lista de todos los formularios registrados
`GET /forms/{formId}`	Obtiene los detalles de un formulario
`GET /forms/{formId}/elements`	Obtiene los elementos de un formulario
`GET /forms/{formId}/data`	Obtiene todos los datos recogidos para un formulario
`GET /forms/{formId}/data?urisToInclude={elementIds}`	Obtiene los datos de los elementos especificados en una lista
`POST /forms/{formId}/analysis/predictor?filter={filter}&filterParams={params}`	Crea un nuevo predictor (clasificador)
`GET /forms/{formId}/analysis/predictor/{predictorId}`	Obtiene los detalles de un predictor
`GET /forms/{formId}/analysis/predictor/{predictorId}/representation`	Obtiene la representación de un predictor
`POST /forms/{formId}/analysis/predictor/{predictorId}/prediction`	Realiza una predicción utilizando un predictor creado

Tabla 4.2: Operaciones disponibles en la API de análisis

La API expone una interfaz HTTP utilizando JSON para la representación de datos, está implementada utilizando Java EE 6 y está desplegada en un servidor JBoss Wildfly 8. Ésta API accede a los datos de ODK Aggregate mediante una

conexión a su base de datos donde los datos son analizados utilizando la suite de aprendizaje de máquina 'Weka'.

Los usuarios (personas u aplicaciones) acceden a las operaciones disponibles en el servicio de análisis de datos realizando invocaciones HTTP, no se provee una interfaz gráfica, debido que escapa a los objetivos del proyecto. La figura 4.1 muestra un esquema general de la arquitectura, incluyendo el módulo de análisis de datos. A continuación se detalla cómo los datos son accedidos e interpretados y cómo se lleva a cabo el análisis.

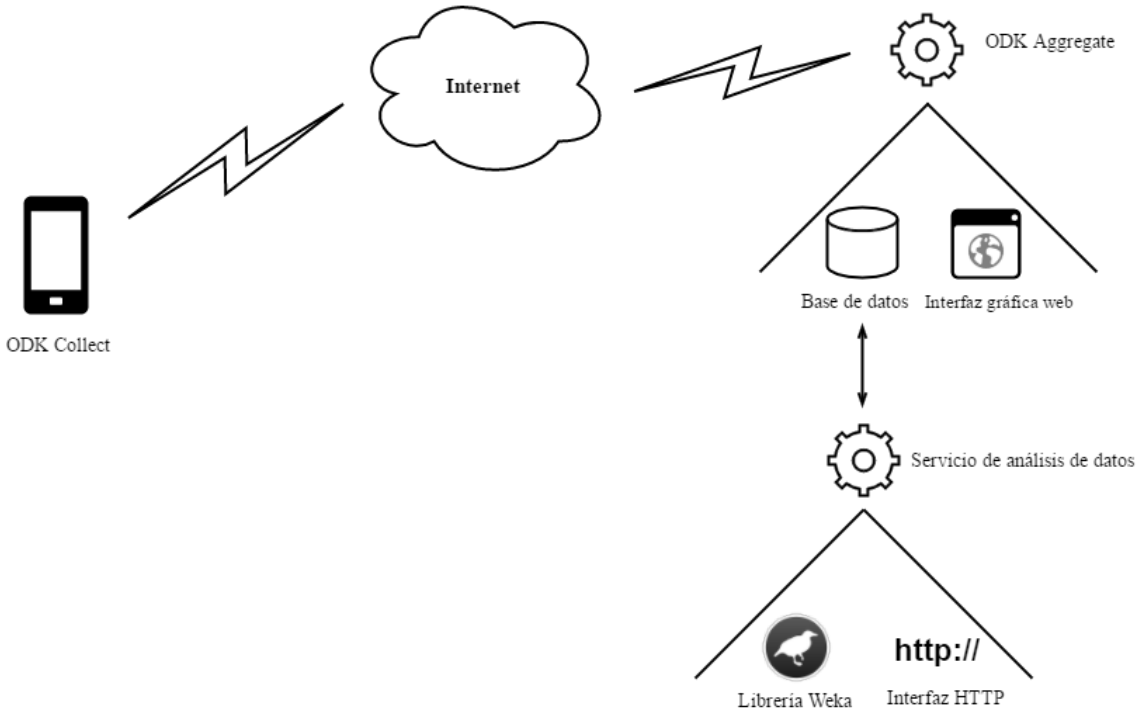


Figura 4.1: Arquitectura general

Estructura de datos

Para que los datos de los formularios sean accedidos desde el proceso de análisis de datos se realiza una conexión TCP al gestor de base de datos utilizado por ODK Aggregate, que puede ser utilizado con MySQL o Postgresql. En esta base de datos residen los formularios registrados así como los datos recogidos en una estructura determinada por ODK Aggregate.

ODK Aggregate utiliza la tabla `_FORM_INFO` para almacenar todos los identificadores de formularios (`form_id`) registrados en el sistema, en otra tabla `_FORM_INFO_SUBMISSION_ASSOCIATION` se almacenan los forms id junto con su número de versión del formulario y un identificador de modelo (`model_id`). Estas tablas se utilizan para mostrar la lista de formularios disponibles en la API REST.

Aggregate almacena los datos utilizando una tabla por cada formulario, y además, creando tablas individuales por cada grupo definido dentro del xform si fuera necesario. Por ejemplo, si se tiene un grupo describiendo el proveedor, otro grupo

describiendo el paciente, y un grupo describiendo la razón de la visita y los tratamientos realizados. En caso que esta información no quepa en una sola tabla, el sistema tratará de dividirla en 3 tablas, una por cada grupo definido en el xform.

La tabla principal para el almacenamiento de datos del formulario finaliza con el nombre `_CORE` o `_COREn`, donde `'n'` es un número. El nombre de la tabla principal se forma mediante `form_id + ''_'' + nombre_grupo + ''_CORE''`, el valor de `form_id` y `nombre_grupo` se comprimen a 64 o menos caracteres, si se tiene un `form_id` muy largo se comprimirá a un valor más corto.

El mapeo de valores a columnas se mantiene en la tabla `_FORM_DATA_MODEL`, para acceder a esta tabla se debe conocer el `model_id` del formulario, esta columna recibe identificadores en el formato UUID (universally unique identifier) [1] . En la misma, las columnas `persist_as_schema_name`, `persist_as_table_name` y `persist_as_column_name` identifican el esquema, el nombre de la tabla y la columna, respectivamente, donde el valor del elemento del formulario será almacenado.

Las columnas `element_name` y `element_type` identifican al nombre de la etiqueta xform y al tipo de elemento del formulario. El anidamiento de los elementos dentro de otros, incluyendo los grupos de repetición se determina mediante un enlace a la tabla `_FORM_DATA_MODEL` vía la columna `parent_uri_form_data_model`, el cual almacena el identificador del elemento padre [Kitd].

Los elementos binarios y de selección múltiple se almacenan en tablas separadas. Los datos de selección múltiple se almacenan de manera que cada registro de la tabla de selección múltiple contenga un enlace de nuevo a la fila de la tabla en la que está asociado.

Una vez que el cliente solicita la operación, se realizan las consultas necesarias en la base de datos para obtener la información requerida. La figura 4.2 muestra un diagrama de base de datos con las tablas utilizadas por ODK Aggregate para el manejo del repositorio de datos de formularios genéricos, si bien, el sistema utiliza además otras tablas para el almacenamiento de configuraciones relacionadas a su uso, se omiten por simplicidad.

Análisis con Weka

Weka es una colección de algoritmos de aprendizaje de máquina para realizar tareas de minería de datos. Los algoritmos pueden ser aplicados directamente a un conjunto de datos utilizando una interfaz gráfica o utilizados desde código Java. Weka contiene herramientas para preprocesamiento de datos, clasificación, regresión, agrupamiento (clustering), reglas de asociación y visualización [Mac].

Para poder utilizar los algoritmos de aprendizaje de máquina, se incluye la librería Java de Weka de manera a poder invocar directamente los algoritmos desde el código fuente del servicio de análisis de datos. Antes de utilizar cualquier algoritmo, los datos se recogen desde las tablas de ODK Aggregate y se transforma a un formato particular utilizado por Weka: ARFF.

Archivo de formato Atributo-Relación: ARFF

[1] UUID es un número generado de 128 bits que para propósitos prácticos es único. La probabilidad que haya un UUID duplicado no es cero, pero es muy cercano a cero [ITU]

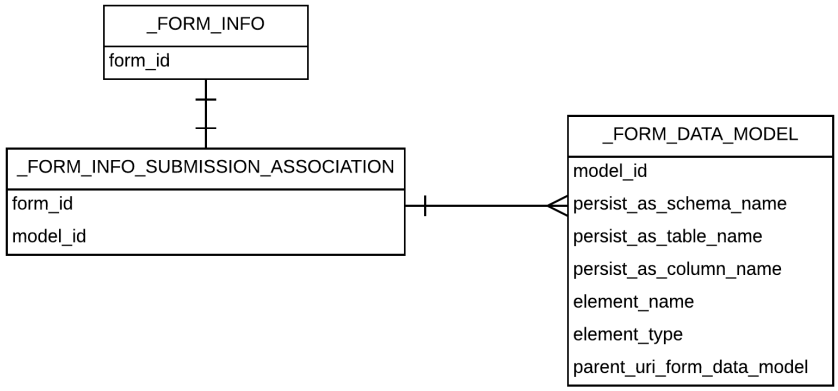


Figura 4.2: Tablas utilizadas por ODK Aggregate para la definición del repositorio de datos

El formato ARFF (Attribute-Relation File Format), es un archivo de texto plano que describe una lista de instancias que comparten atributos. Se compone de dos secciones: la cabecera seguida del conjunto de datos. La cabecera del archivo ARFF contiene el nombre de la relación, una lista de atributos y sus tipos asociados. Por ejemplo:

```
% 1. Title: Iris Plants Database
%
% 2. Sources:
% (a) Creator: R.A. Fisher
% (b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
% (c) Date: July, 1988
%

@RELATION iris
@ATTRIBUTE sepallength NUMERIC
@ATTRIBUTE sepalwidth NUMERIC
@ATTRIBUTE petallength NUMERIC
@ATTRIBUTE petalwidth NUMERIC
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}
```

Los datos del archivo ARFF se representan a continuación:

```
 @DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
```

```
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Sección cabecera Contiene la declaración de la relación y los atributos.

@relation Define el nombre de la relación en la primera línea del archivo ARFF. El formato de la sentencia es `@relation <nombre>`.

@attribute Declara los atributos del conjunto de datos, una a continuación de otra en cada línea. El orden en el que los atributos son declarados indica la posición de la columna en la sección de datos del archivo. Por ejemplo, si un atributo es el tercero declarado, entonces Weka espera que sus valores se encuentren en la tercera columna de la sección de datos.

El formato para la sentencia es `@attribute <nombre> <tipo-de-dato>`. El tipo de dato puede ser uno de:

- *numeric*: Números reales o enteros.
- *<especificación-nominal>*: Lista de valores nominales encerrados en llaves y separados por coma. Ej: `{<valor-nominal1>, <valor-nominal2>, ...}`.
- *string*: Valores textuales.
- *date*: Valores de fecha/hora.

Sección de datos Contiene la declaración de inicio de los datos y las instancias de los mismos.

@data Indica el inicio del segmento de datos del archivo. El formato es: `@data`.

Instancias de datos Cada instancia se representa en una sola línea, los valores de los atributos de la instancia se separan por comas y deben aparecer en el orden en el que fueron declarados los atributos en la cabecera del archivo.

Algoritmos

Las operaciones de minería de datos disponibles en la API Rest son las siguientes:

Crear un modelo de clasificación Dado un formulario y los atributos a utilizar, se entrena un algoritmo para crear un modelo de clasificación que podrá ser utilizado para realizar predicciones de acuerdo a los parámetros seleccionados. Al modelo de clasificación también se lo denomina predictor. El algoritmo de clasificación utilizado se denomina *J48*. *J48* es una implementación opensource del algoritmo de árboles de decisión C4.5 de Ross Quilan escrita en Java [KW09]. Una vez que el modelo de clasificación es creado, se lo evalua con el método de validación cruzada de 10 iteraciones (10 fold cross-validation) y se retornan métricas de evaluación del modelo de clasificación. Las evaluaciones retornadas son:

- Número de clases
- Nombre de las clases
- Número de instancias
- Matriz de confusión
- Exactitud
- Tasa de error
- Cantidad de instancias clasificadas correctamente
- Porcentaje de instancias clasificadas correctamente
- Cantidad de instancias clasificadas incorrectamente
- Porcentaje de instancias clasificadas incorrectamente
- Cantidad de instancias no clasificadas
- Porcentaje de instancias no clasificadas
- Por cada clase:
  - Nombre de la clase
  - Sensibilidad
  - Especificidad
  - Precisión
  - Exhaustividad
  - Medida f
  - Área bajo la curva ROC

Ejemplo: Crear un predictor utilizando *J48*

Petición

```
POST /forms/md5:6016c342e4bcbdca9e479d69f96f3652/analysis/predictor
{
   "uris":[
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000007)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000008)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000014)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000015)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000016)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000018)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000019)",
      "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000020)"
   ],
   "classUri":"elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000022)",
   "classifier":"J48"
}
```

Respuesta

```
HTTP/1.1 201 Created
Location: /forms/md5:6016c342e4bcbdca9e479d69f96f3652/analysis/predictor/1
{
  "numClasses": 2,
  "classNames": [ "captura_negativa",
                  "captura_positiva" ],
  "confusionMatrix": {
    "titleRow": [ "a", "b" ],
    "classColumn": [ "a = captura_negativa",
                     "b = captura_positiva" ],
    "matrix": [ [ 138.0, 0.0 ],
                [ 10.0, 0.0 ] ]
  },
  "correctlyClassifiedInstances": 138.0,
  "pctCorrectlyClassifiedInstances": 93.24324324324324,
  "incorrectlyClassifiedInstances": 10.0,
  "pctIncorrectlyClassifiedInstances": 6.756756756756757,
  "unClassifiedInstances": 0.0,
  "pctUnClassifiedInstances": 0.0,
  "numInstances": 148.0,
  "accuracy": 0.9324324324324325,
  "errorRate": 0.06756756756756754,
  "detailedAccuracyByClass": [
    {
      "sensitivity": 1.0,
      "specificity": 0.9,
      "precision": 0.9324324324324325,
      "recall": 1.0,
      "fMeasure": 0.965034965034965,
      "areaUnderROC": 0.49075462268865566,
      "class": "captura_negativa"
    }, {
      "sensitivity": 0.0,
      "specificity": 0.0,
      "precision": 0.0,
      "recall": 0.0,
      "fMeasure": 0.0,
      "areaUnderROC": 0.4967741935483871,
      "class": "captura_positiva"
    }
  ]
}
```

Utilizar un modelo de clasificación creado Una vez entrenado el modelo de clasificación éste se podrá utilizar posteriormente para realizar predicciones. Éste modelo está asociado a un formulario y la lista de atributos elegidos, motivo por el cual solo se podrá utilizar para dicha combinación de formularios y atributos. Una vez realizada la predicción, se retorna el valor de la etiqueta de clase.

Ejemplo: Utilizar el clasificador creado para realizar una predicción.

Petición

```
POST /forms/md5:6016c342e4bcbdca9e479d69f96f3652/analysis/predictor/0/prediction
{
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000007)":"Campo Largo",
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000008)":2,
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000014)":2,
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000015)":3,
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000016)":2,
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000018)":"Tronco",
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000019)":"Paja",
   "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000020)":"Tierra"
}
```

Respuesta


```
{
  "classification": "Captura negativa"
}
```

Capítulo 5

Caso de estudio y resultados

En este capítulo se describe el caso de estudio en donde se hizo uso de la solución propuesta, se presentan los resultados y una evaluación de los mismos.

5.1. Caso de estudio

5.1.1. Antecedentes

El Programa Nacional de Control de la Enfermedad de Chagas impulsa la realización de encuestas a hogares de comunidades de la región del Chaco del Paraguay con el fin de recoger variables socio-económicas, comportamientos, costumbres y creencias sobre la enfermedad de Chagas y de esta forma prevenir la mortalidad y disminuir las pérdidas socioeconómicas ocasionadas por la enfermedad [Arr+14]. Actualmente estas encuestas son ejecutadas por el Centro para el Desarrollo de la Investigación Científica (CEDIC), quienes son los encargados de la preparación, logística, recolección y procesamiento de las mismas. Los formularios están impresos en papel, y son transportados hasta el lugar del estudio para la recolección de los datos, donde una vez recogidos son enviados de vuelta a las oficinas de CEDIC para su digitalización utilizando programas como Microsoft Access.

5.1.2. Toma de datos

Se tuvo la necesidad de realizar una encuesta a comunidades de origen indígena en los municipios de Teniente Irala Fernández, departamento de Presidente Hayes y Loma Plata, departamento de Boquerón utilizando los formularios que se describen en la siguiente sección. La encuesta se realizó desde el día 14 de octubre del 2014 hasta 19 de octubre inclusive.

5.1.3. Formularios

La encuesta consistió en 4 formularios utilizados por encuestadores:

Formulario	Contenido	Cantidad de preguntas
Cuestionario domiciliar	▪ Datos de la vivienda ▪ Localización GPS ▪ Tipo estructura de la vivienda ▪ Prevalencia de triatominos	26
Encuesta de hogares	▪ Datos del hogar ▪ Conformación del hogar ▪ Ocupación y empleo del jefe del hogar ▪ Movimiento de la población ▪ Datos detallados de la vivienda y el peridomicilio ▪ Datos de animales que se poseen ▪ Ambiente circundante ▪ Controles de salud ▪ Fotos de la vivienda	98
Apéndice Chagas	▪ Rociado de la casa ▪ Creencias y conductas relacionadas a la vinchuca ▪ Visualización de vinchucas en la casa ▪ Eliminación de las vinchucas ▪ Información sobre la enfermedad de Chagas	49

Apéndice de participación social	▪ Población perteneciente (criolla o indígena) ▪ Participación en organizaciones sociales ▪ Actividades sociales ▪ Relacionamiento con la comunidad	22

Tabla 5.1: Formularios utilizados en la encuesta.

5.1.4. Propuesta

Se propuso realizar las encuestas utilizando el modelo presentado en la sección 4.1, utilizando tabletas para la recolección de datos y enviando los resultados cuando se disponga de una conexión a internet. Una vez obtenidos los datos realizar el análisis de los mismos evaluando la herramienta.

5.1.5. Construcción de formularios

Los formularios fueron provistos en papel por parte de CEDIC y fueron digitalizados utilizando ODK Build y planillas electrónicas en formato XLS. La distribución fue la siguiente:

- ODK Build:
  - Cuestionario domiciliar
  - Apéndice Chagas
- Planillas electrónicas:
  - Encuesta de hogares
  - Apéndice de participación social

En el caso de ODK Build, una vez que el formulario fue diseñado, se descargó su definición en formato XForm en un archivo XML que luego fue cargado a una instancia de ODK Aggregate, o directamente al dispositivo móvil. En el caso de las planillas electrónicas, se utilizó un convertidor de XLS a XForm disponible en el sitio web de Open Data Kit (`http://opendatakit.org/xiframe/`). Alternativamente se podrían haber utilizado otros convertidores que no necesitan una conexión a internet activa.

	type	name	label::Español
2	start	start	
3	end	end	
4	deviceid	deviceid	
5	text	ultimo_rociado	¿Recuerda cuándo fue la última vez que se roció su casa?
6	select_one si_no_indiferente	gusto_rociado	¿Le gustó que se hiciera el rociado de su casa?
7	select_one si_no_indiferente	gusto_trabajo	¿Le gustó la manera en que se hizo el trabajo?
8	text	porque_gusto_trabajo	¿Por qué?
9	begin group	creencias_conductas_vinchuca	Creencias y conductas relacionadas con la vinchuca
10	begin group	preguntas_animales_bichos	Algunas preguntas sobre los animales y bichos de la zona
11	text	mas_molestos	¿Cuáles son los más molestos?
12	text	mas_peligrosos	¿Cuáles son los más peligrosos?
13	text	mas_dificiles_matar	¿Cuáles son los más difíciles de matar?
14	end group		
15	select_one si_no	reconoce_vinchuca	¿Reconoce la vinchuca cuando la ve?
16	integer	cantidad_clase_vinchuca	¿Cuántas clases/tipos de vinchuca conoce?
17	text	cuales_vinchucas	¿Cuáles son?
18	text	mas_peligrosa	¿Cuál es la más peligrosa?
19	select_one si_no_nsnc	preocupa_vinchucas_casa	¿Le preocupa que en su casa pueda haber vinchucas?
20	text	porque_preocupa	¿Por qué le preocupa?
21	text	porque_no_preocupa	¿Por qué no le preocupa?
22	select_one si_no_nsnc	hay_vinchucas_casa	¿Hoy en día hay vinchucas en su casa?
23	select_one si_no_nsnc	encontrado_vinchucas_casa	¿Ha encontrado alguna vez vinchucas en su casa?
24	begin group	si_ha_encontrado_vinchucas_casa	Sí se ha encontrado vinchucas en la casa
25	text	ultimo_encuentro_vinchucas	¿Cuándo fue la última vez que encontró vinchucas?
26	text	que_hizo_ver	¿Qué hizo al verlas?
27	text	razon_vinchucas_casa	¿Por qué cree que había vinchucas en su casa?
28	select_one si_no_nsnc	cree_volver	¿Cree que van a volver?
29	text	por_que_volvera	¿Por qué?
30	text	porque_cree_vinchucas_casa	¿Por qué cree que ya no hay vinchucas en su casa?
31	end group		

Figura 5.1: Formulario digitalizado en planilla electrónica

5.1.6. ODK Aggregate

Se realizó una instalación local de ODK Aggregate en el clúster de investigación de la Facultad Politécnica - UNA. Aggregate se distribuye en forma de un empaquetado WAR (Web Application Archive) que debe ser desplegado en un contenedor de servlets.

Software utilizado:

- Sistema operativo: Ubuntu 11.04 64-bit
- Contenedor de servlets: Apache Tomcat 6.0.43
- JVM: Oracle Java 1.7.0_17
- Base de datos: PostgreSQL 9.3.4

Hardware utilizado:

- Procesador: Intel(R) Xeon(R) CPU E5530 @ 2.40GHz
- Memoria: 16 GB
- Almacenamiento: 3 TB

Una vez desplegada la aplicación web, se crearon los usuarios y sus permisos. Además, se cargaron las definiciones de los formularios lo que posibilitó obtenerlos remotamente usando la dirección URL del sitio web.

5.1.7. Equipos móviles

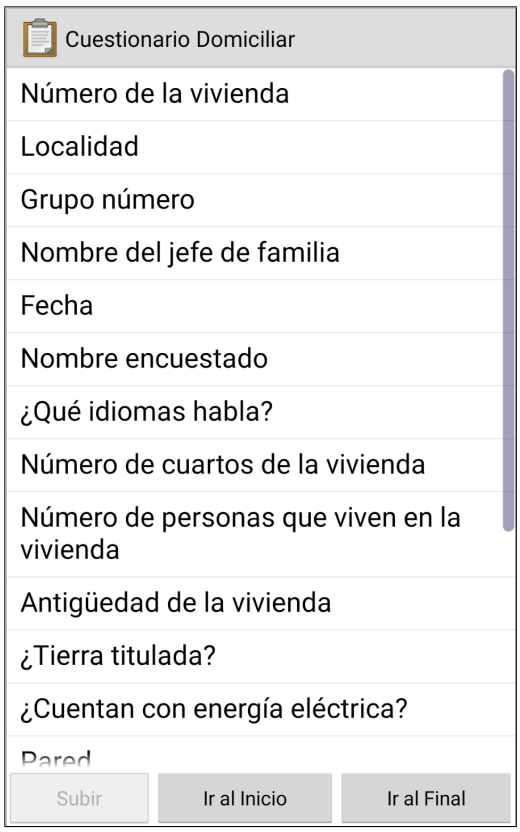
Se utilizaron cuatro Samsung Galaxy Tab 4 con las siguientes características

- Pantalla: 10.1 pulgadas, 1280 x 800 píxeles
- CPU: 1.2 GHz Quad Core
- Sistema Operativo: Android 5.0 Lollipop
- Cámara: 3.15 Megapíxeles
- Comunicación: WiFi a/b/g/n, GSM/HSPA/LTE
- GPS: A-GPS, GLONASS

Los dispositivos fueron equipados con tarjetas SIMs de una compañía telefónica con cobertura en la mayoría de las áreas en donde se realizará la encuesta, de esta manera las tablets contaron con conexión móvil de datos.

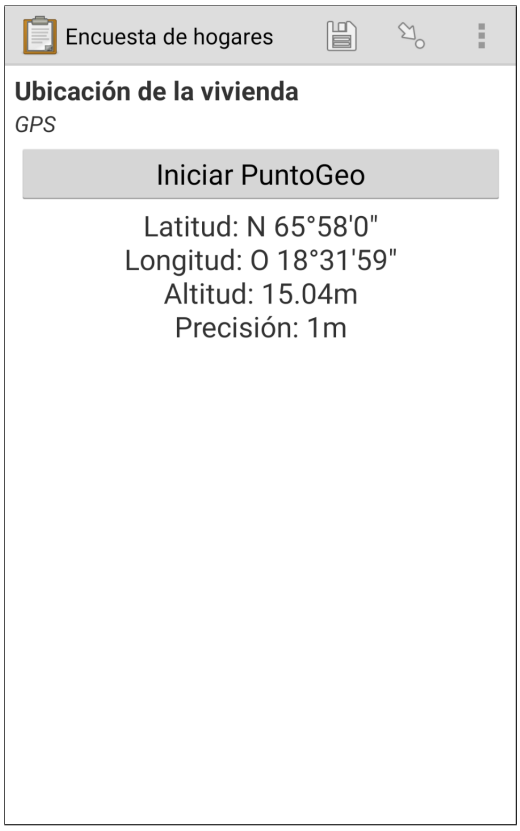
5.1.8. Configuración de equipos

En cada tablet se instaló ODK Collect versión 1.4.7 vía Google Play. Se verificó el funcionamiento correcto de la aplicación y posteriormente fue configurada con la URL de ODK Aggregate con sus respectivas credenciales. Se obtuvieron las definiciones de los formularios y se realizaron tomas de datos de prueba. En la figura 5.2 se muestran los formularios en el dispositivo.

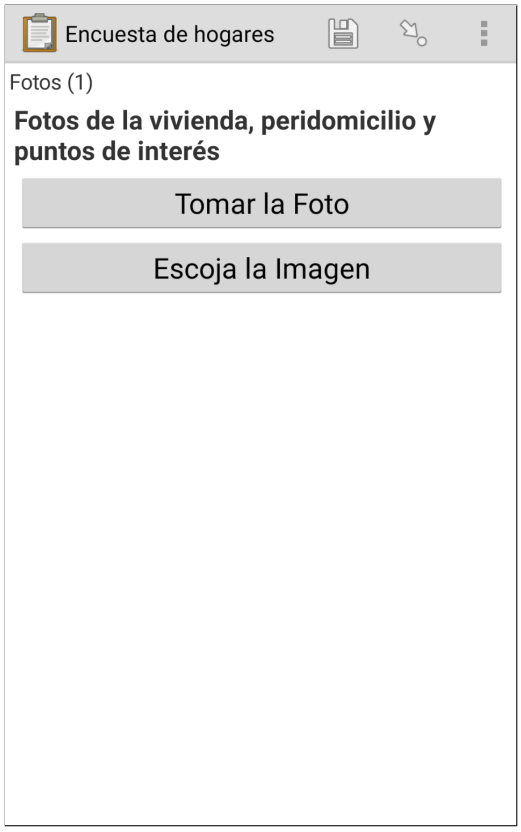


(a) Lista de preguntas a realizar

(b) Una pregunta de selección múltiple.



(c) Toma de localización



(d) Toma de fotos

Figura 5.2: Formularios cargados en el dispositivo y listos para recoger información.

5.1.9. Capacitación

Una semana antes del inicio de la recolección de datos se realizó una capacitación con las personas que utilizarían el software para la toma de datos en campo. Se informó de cómo funcionaría el proceso y cómo utilizar la aplicación para recoger los datos. A continuación, se realizó una demo de toma de datos por parte de los futuros encuestadores, que consistió en salir a recoger datos de prueba en los domicilios situados en los alrededores del sitio de la capacitación.

5.2. Resultados

5.2.1. Toma de datos

Para la recolección de datos, cada encuestador contó con una tablet para realizar el trabajo. Éste utilizó los cuatro formularios por cada hogar, a cada vivienda se le asignó un identificador único. Ésta clave se utilizó para luego identificar los datos pertenecientes a un hogar entre los formularios completos. En total se encuestaron 196 casas, distribuidas como se muestra en la tabla 5.2.

Cantidad	Localidad	Departamento	Etnia
36	Jope	Boquerón	Nivaclé
32	Betania	Boquerón	Nivaclé
31	Tiberia	Boquerón	Nivaclé
17	Martillo	Presidente Hayes	Angaité
17	Karandilla	Presidente Hayes	Angaité
63	12 de Junio	Presidente Hayes	Angaité

Tabla 5.2: Casas encuestadas por localidad y etnia

Debido a las distancias que se debían recorrer y a la dificultades del terreno no se podían encuestar la totalidad de las casas en la misma jornada. Se requirieron en total 8 días para tomar los datos en todos los hogares. Al final de cada jornada, cada tablet era conectada a internet y los datos se enviaban al servidor ODK Aggregate que se dispuso para el efecto en el clúster de la FPUNA (sección 5.1.6). En el otro extremo, los datos enviados eran verificados para asegurar su correcta recepción; en este punto la información pudo ser inspeccionada y evaluada de tal manera que pudo darse una retroalimentación con las personas que se encargan de la recolección de los datos.

5.2.2. Visualización

Datos

Los datos se pudieron visualizar en forma tabular, donde cada columna está nombrada con el identificador asignado a cada pregunta del formulario, como se muestra

en la figura 5.3. En la imagen se muestran los metadatos capturados automáticamente por ODK Collect, los mismos corresponden a la fecha, hora del inicio y fin de la toma de datos para dicha instancia, y el identificador del dispositivo con el que fue realizado. Se pudieron aplicar filtros correctamente para mostrar datos en base a un criterio elegido. Además, la función de exportación a CSV hace que los datos puedan ser utilizados con software de planillas electrónicas.

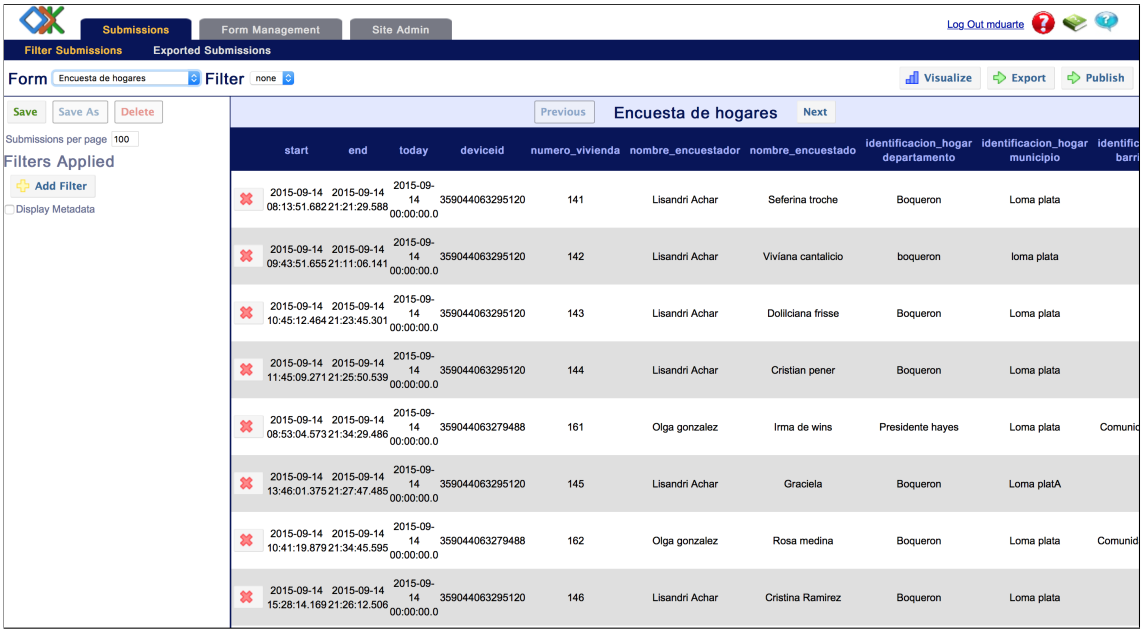


Figura 5.3: Formulario de Cuestionario Domiciliar: Visualización de datos en forma tabular con ODK Aggregate

Mapas

En el formulario de encuesta de hogares se captura la información de la localización de cada casa. Junto con la información de la posición, se obtiene además la altitud del terreno y la exactitud de la posición en metros. Se pudo obtener el mapa de todos los puntos de recogida de datos, como se muestra en la Figura 5.4. Además, al seleccionar un punto se pueden mostrar fotos y datos relevantes al marcador elegido. La Figura 5.6 muestra el mapa en donde se muestra información referente a un punto seleccionado, la Figura 5.5 muestra un mapa exportado a Google Maps, lo que permite compartir el mapa con terceros sin necesidad de crear una cuenta en ODK Aggregate.

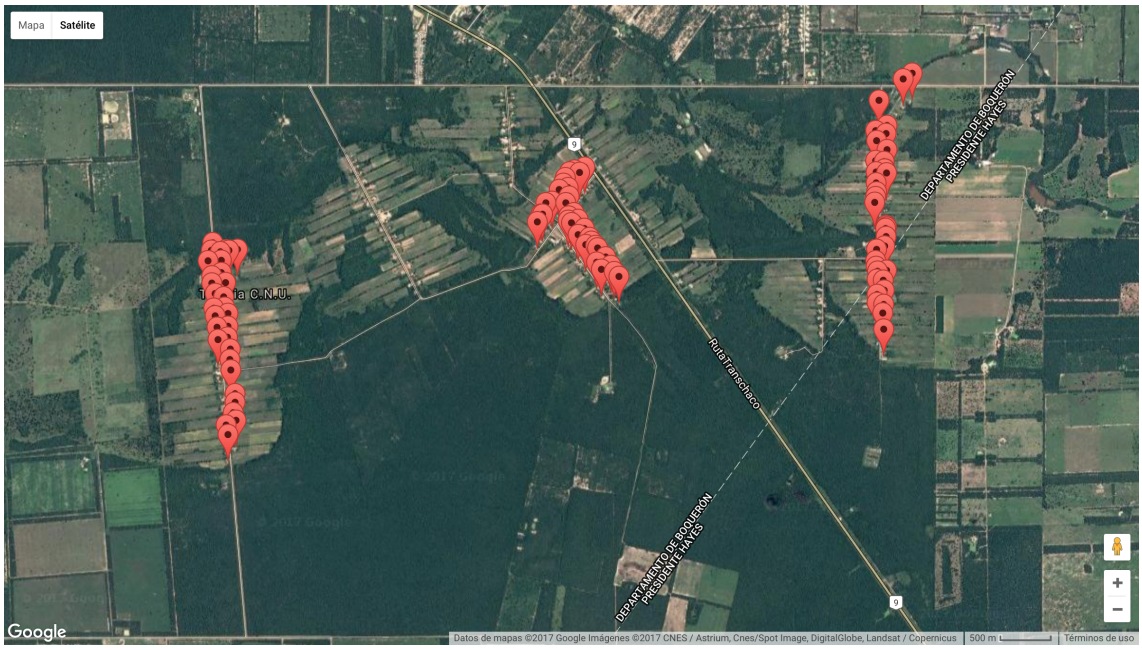


Figura 5.4: Localizaciones de todas las encuestas realizadas.

Figura 5.5: Mapa exportado a Google Maps.

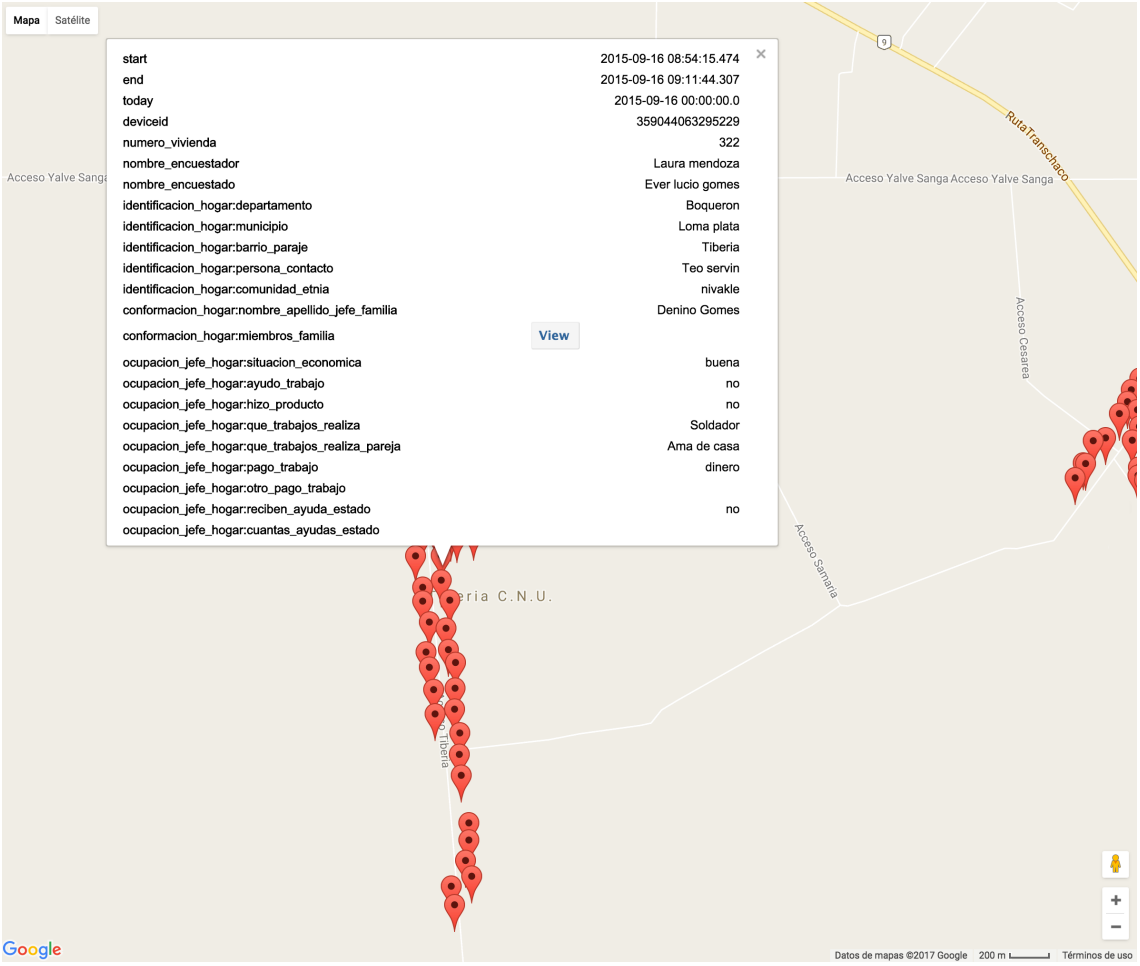


Figura 5.6: Mapa generado por ODK Aggregate.

Gráficos

ODK Aggregate permite la generación de gráficos de tipo torta y barras horizontales. Cada gráfico permite contar las ocurrencias de una variable seleccionada, o la suma de una variable numérica versus una variable discreta. Para las encuestas, por ejemplo, se generaron los siguientes gráficos:

- Cantidad de ocurrencias de tipos de pared (gráfico de barra), Figura 5.7.
- Porcentaje de ocurrencias de tipos de pared (gráfico de torta), Figura 5.8.
- Suma de la cantidad de personas por tipo de tierra (gráfico de barra), Figura 5.9.

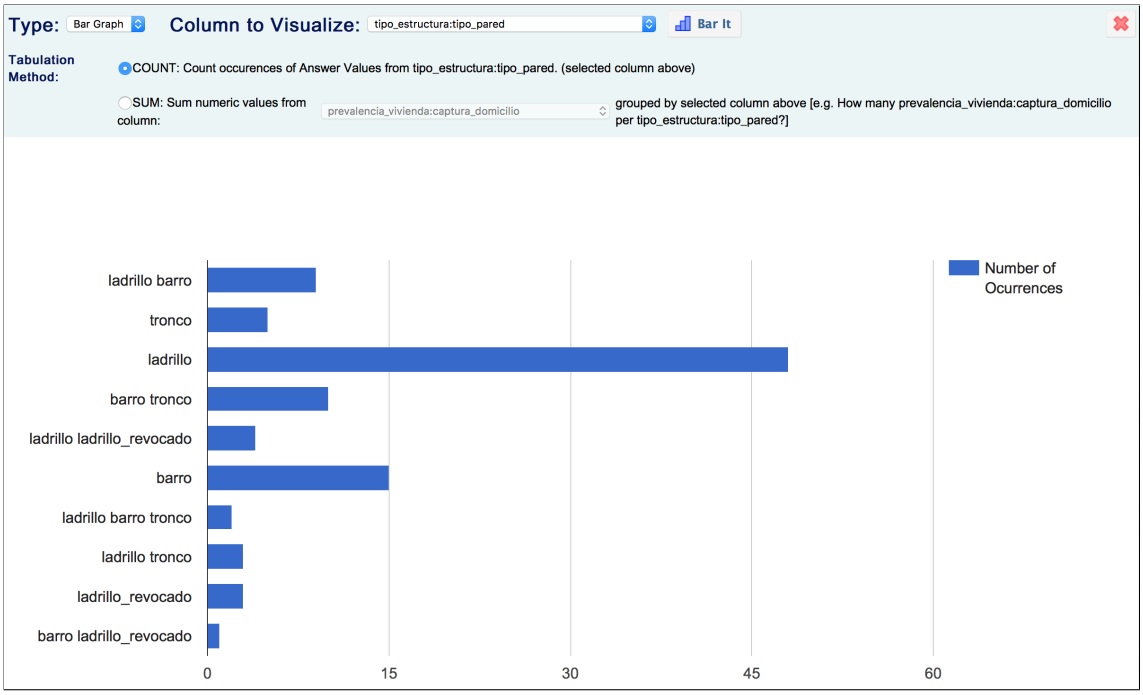


Figura 5.7: Cantidad de ocurrencias de tipos de pared.

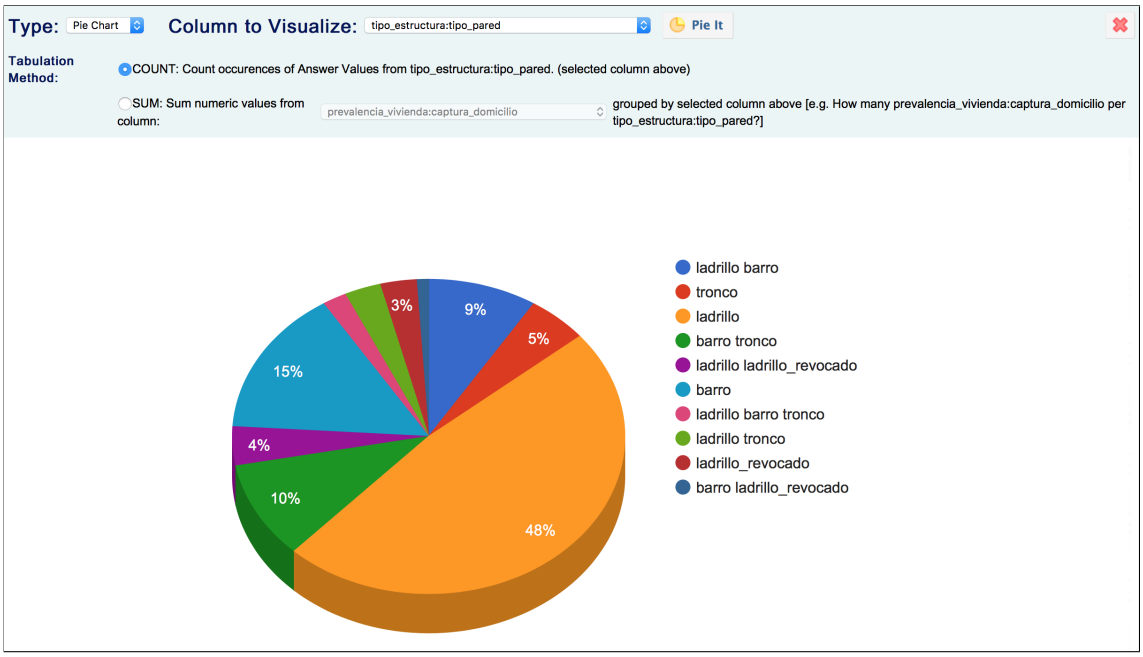


Figura 5.8: Porcentaje de ocurrencias de tipos de pared.

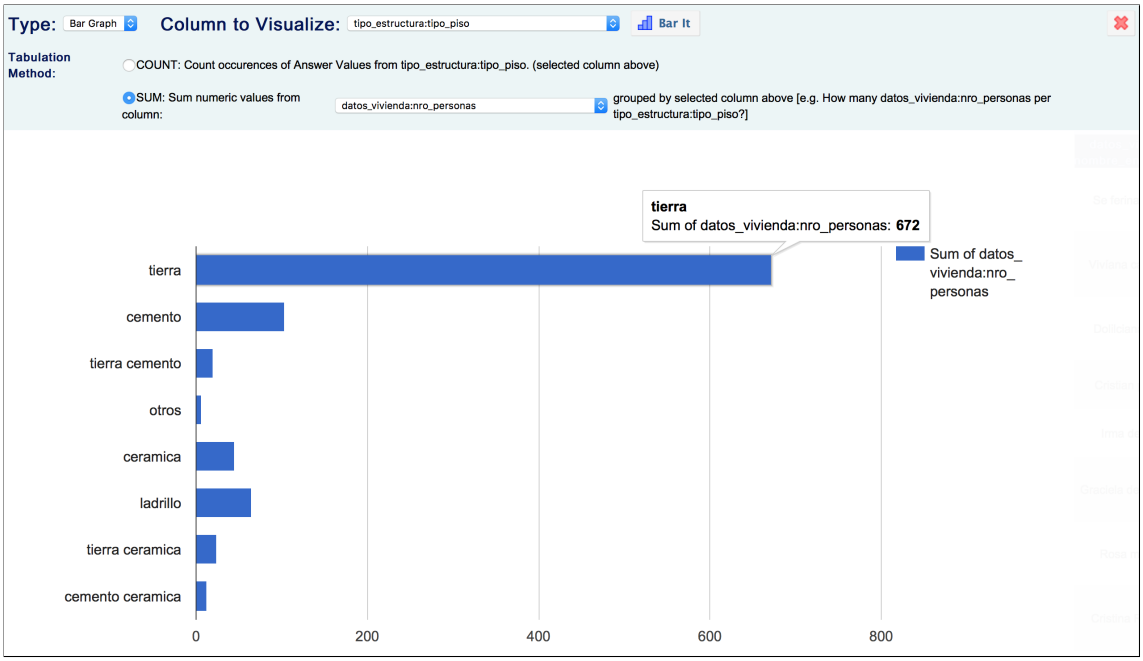


Figura 5.9: Suma de la cantidad de personas por tipo de tierra.

5.2.3. Análisis

El objetivo del análisis es construir modelos de clasificación mediante los algoritmos y técnicas descriptos en el capítulo 3, compararlos, evaluar sus desempeños y tratar de extraer conclusiones a través de los mismos.

Para realizar el análisis de datos se escogió la encuesta "Cuestionario Domiciliar". Se podrían utilizar también las demás encuestas, sin embargo, la elegida recoge datos acerca de un fenómeno importante: la presencia de vinchucas en el domicilio y en el peridomicilio. En la tabla 5.3 se listan las preguntas, que se utilizarán como variables, pertenecientes a la encuesta seleccionada.

Identificador	**Descripción**	**Categorías**
numero_vivienda	Número de la vivienda	Numérica
localidad	Localidad	Texto
grupo_numero	Grupo número	Numérica
nombre_jefe	Nombre del jefe de familia	Texto
fecha	Fecha de la encuesta	Texto
gps	Localización GPS	Numérica
nombre_encuestado	Nombre del encuestado	Texto
idiomas_habla	Idiomas que habla	Español Guaraní Otros
tierra_titulada	La tierra en donde habita está titulada	Si/No
anho_titulacion	Año de titulación del terreno	Numérica

cantidad_hectareas	Cantidad de hectáreas del terreno	Numérica
nro_cuartos	Número de cuartos de la vivienda	Numérica
nro_personas	Número de personas en la vivienda	Numérica
antiguedad	Antigüedad de la vivienda	Numérica
cuentan_energia_electrica	Cuenta con energía eléctrica	Si/No
tipo_pared	Tipo de pared	Ladrillo Barro Tronco Pared francesa Ladrillo revocado Pared francesa revocada
tipo_techo	Tipo de techo	Paja Tejas Zinc Tronco Otros
tipo_piso	Tipo de piso	Tierra Cemento Ladrillo Cerámica Otros
captura_domicilio	Captura de triatominos en el domicilio	Captura Positiva Captura Negativa
tipo_captura_domicilio	Tipo de captura en el domicilio	Adultos Ninfas Huevos embrionados Huevos eclosionados Otros
lugar_captura_domicilio	Lugar captura en el domicilio	Paredes Piso Techo Camas Ropas Otros
captura_peridomicilio	Captura de triatominos en el peridomicilio	Captura Positiva Captura Negativa
tipo_captura_peridomicilio	Tipo de captura en el peridomicilio	Adultos Ninfas Huevos embrionados Huevos eclosionados Otros

lugar_captura_peridomicilio	Lugar captura en el peridomicilio	Gallinero Depósito Galpón Corral Materiales acumulados Otros
bservaciones	observaciones	Texto

Tabla 5.3: Variables utilizadas en el cuestionario domiciliar

Para este conjunto de datos utilizaremos como etiqueta de clase a la variable "captura_domicilio", de esta manera buscaremos crear un modelo que determinará si tendremos una captura positiva o negativa en el domicilio utilizando las demás variables capturadas. Definimos como tupla positiva al valor "captura_positiva". La distribución de la etiqueta de clase se muestra en la tabla 5.4.

Tipo de captura	**Cantidad**
Positiva (captura_domicilio = captura_positiva)	10
Negativa (captura_domicilio = captura_negativa)	138

Tabla 5.4: Distribución del tipo de captura.

Como paso previo procederemos a remover variables que no tienen relevancia o no hacen a la presencia de vinchucas:

- Número de vivienda (nro_vivienda)
- Número de grupo (grupo_nro)
- Nombre del jefe de familia (nombre_jefe)
- Fecha de la encuesta (fecha)
- Localización GPS (gps)
- Nombre del encuestado (nombre_encuestado)
- Observaciones (observaciones)
- Idiomas que habla (idiomas_habla)

La variable "Cantidad de heactáreas" (cantidad_hectareas) no fue incluida debido a que presentaba datos anómalos, con muchos valores entre 4000 y 8000 Ha, lo que hacía que la media de dicha variable fuera 2893 Ha y la desviación estándar 3740 Ha.

A continuación se presentan los resultados de los algoritmos aplicados. Para la evaluación de los modelos se utilizará la nomenclatura introducida en la sección 3.2.

Clasificación

Se utiliza el clasificador J48 directamente sobre el conjunto de datos.

Evaluación
Método de evaluación: validación cruzada de 10 iteraciones.

- $Exactitud = 0{,}926$
- $Error = 0{,}074$
- $Sensibilidad = 0{,}1$
- $Especificidad = 0{,}986$
- $Precisión = 0{,}333$
- $Exhaustividad = 0{,}1$
- $F = 0{,}154$
- $Área\ \ ROC = 0{,}655$

Matriz de confusión

		Clase predicha		
		a	b	Total
Clase real	a = captura_positiva	1	9	10
	b = captura_negativa	2	136	138
				148

Árbol de decisión

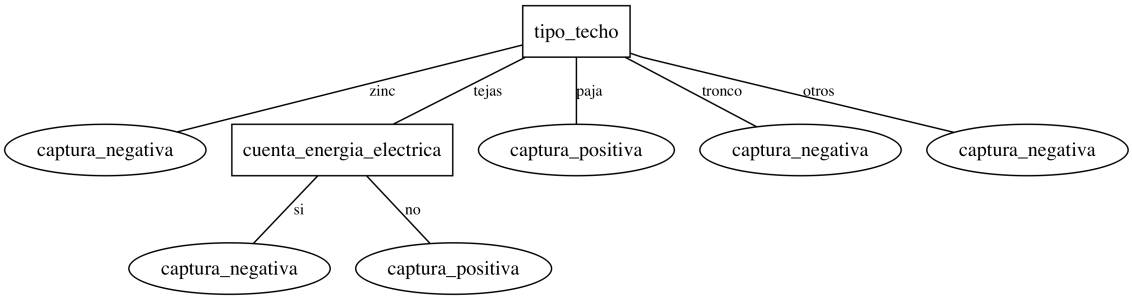


Figura 5.10: Árbol de decisión resultante

Interpretación de resultados
Como se puede observar en los resultados de 5.2.3 obtuvimos las métricas necesarias para evaluar el desempeño de nuestro clasificador. Vemos que obtenemos un valor de *sensibilidad* de 0,1, que se refiere a la tasa de reconocimiento de verdaderos

positivos, se considera que a medida que este valor esté cercano a 1 el clasificador posee un mejor rendimiento para identificar verdaderos positivos. Esto significa que el modelo de clasificación resultante no realiza un buen trabajo identificando capturas positivas.

Si observamos la matriz de confusión obtenida, podemos comprobar que la proporción de "captura_negativa" supera en más de 10 veces a la cantidad de "captura_positiva", además, el clasificador clasifica la mayoría de las tuplas como negativas, lo que lo hace obtener una buena exactitud pero ignorando casi por completo las tuplas positivas. Estos indicios nos llevan a concluir que este problema puede ser tratado como un conjunto con clases desbalanceadas [LD13]. A continuación se realizará un sobremuestreo del conjunto de datos para abordar el problema de las clases desbalanceadas.

Utilizamos el algoritmo SMOTE (algoritmo 2) para realizar el sobremuestreo de clase a nuestro conjunto de datos. SMOTE acepta como parámetro el porcentaje de instancias sintéticas a crear, teniendo como valor por defecto 100 %. Realizamos sucesivos sobremuestreos de clase con SMOTE aplicando luego el clasificador J48 al conjunto de datos resultantes, en cada sobremuestreo se incrementaba el valor de dicho porcentaje en 100 %, empezando por 100 %. En cada resultado obtenido hemos verificado el desempeño del clasificador con las métricas resultantes. Según [Cha+02] una vez que se han utilizando técnicas de sobremuestreo y submuestreo es conveniente comparar los modelos de clasificación resultantes utilizando el área ROC. El mejor resultado se encontró con un sobremuestreo del 1000 %, lo que significa que se crearon 100 instancias sintéticas de la clase "captura_positiva":

Evaluación
Método de evaluación: validación cruzada de 10 iteraciones.

- $Exactitud = 0{,}903$
- $Error = 0{,}097$
- $Sensibilidad = 0{,}982$
- $Especificidad = 0{,}841$
- $Precisión = 0,831$
- $Exhaustividad = 0{,}982$
- $F = 0{,}9$
- $Área\ \ ROC = 0{,}929$

Matriz de confusión

		Clase predicha		
		a	b	Total
Clase real	a = captura_positiva	108	2	110
	b = captura_negativa	22	116	138
				248

Árbol de decisión

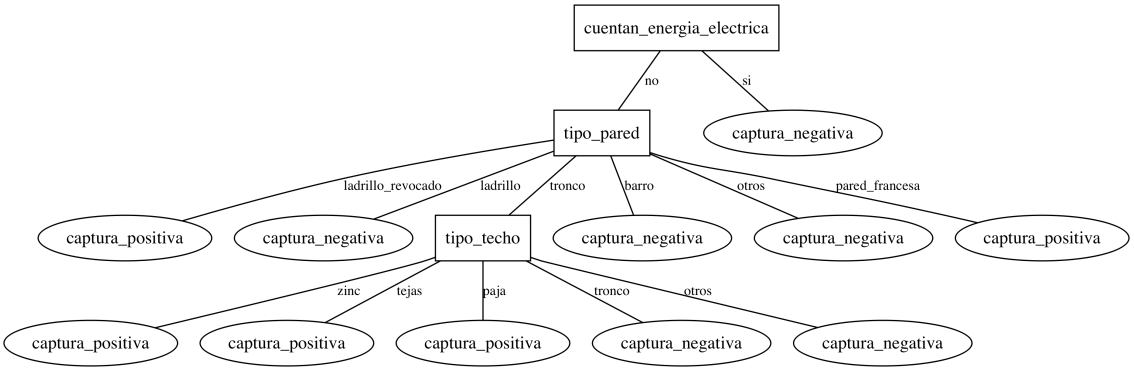


Figura 5.11: Árbol de decisión luego de aplicar SMOTE

Se puede observar que los árboles 5.10 y 5.11 son similares en forma. Sin embargo, los resultados al generar instancias sintéticas de la clase minoritaria con SMOTE ofrecen un mayor detalle en el árbol de decisión, añadiendo el atributo "tipo_pared" como determinante para clasificar una captura como positiva. Al comparar las métricas obtenidas con el clasificador J48 sobre el conjunto de datos pre y post aplicación de SMOTE, debemos destacar que la sensibilidad de la clase minoritaria ("captura_positiva") tiene un aumento destacable, que pasa de un valor de 0,1 a 0,982, lo cual indica que el clasificador tiene un mejor desempeño detectando tuplas positivas luego de la aplicación de SMOTE, tal y como también lo indica el valor del área ROC que pasa de 0,655 a 0,929. Podemos concluir que luego de la aplicación de SMOTE para el sobremuestreo de datos se obtienen mejores métricas de evaluación para determinar el desempeño del clasificador.

5.3. Evaluación

A continuación se evaluarán los resultados obtenidos mediante la aplicación de las herramientas y metodologías propuestas en este trabajo.

5.3.1. Recolección de datos

Construcción de formularios

La herramienta de creación de formularios ODK Build resultó limitada para la creación de formularios complejos. La dificultades encontradas fueron a la hora de crear bifurcaciones condicionales para el formulario y grupos de repetición, características requeridas para dos de las encuestas a realizar en el Chaco. Debido a esto se decidió crear dichos formularios utilizando planillas electrónicas, las cuales poseen la desventaja de necesitar una estructura concreta y utilización de palabras clave para su correcta interpretación. Este último requerimiento puede ser una limitante para la facilidad de uso del usuario final, por lo cual se recomienda utilizar ODK

Build en el caso que el formulario no posea bifurcaciones condicionales ni grupos de repetición.

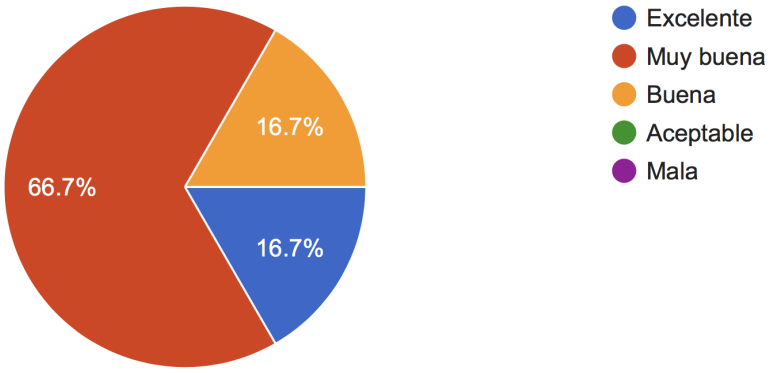
Recolección

La suite de recolección de datos Open Data Kit se destacó por su confiabilidad a la hora de recoger los datos en campo, logrando que no hayan pérdidas de información, los cuales fueron recibidos correctamente cuando fueron enviados. Las opciones de enriquecimiento de la información mediante datos multimedia y GPS fueron utilizados para generar mapas identificando los lugares en donde se recogieron los datos. El componente servidor respondió correctamente mostrando una disponibilidad del 100 % durante los periodos de prueba y recolección de datos. Destacamos además la facilidad de uso de la herramienta, en donde los encuestadores necesitaron solamente una demostración del uso de los formularios electrónicos para poder utilizarlos luego en el campo. Al momento de revisar los datos recogidos se encontraron errores en la entrada de datos, los cuales en su mayoría se pudieron corregir posteriormente. A manera de evitar futuros casos de errores en la entrada de datos se concluyó que los campos deben tener valores cerrados, o en su defecto, validaciones de tipos de datos y de rango, tratando de evitar las entradas de datos totalmente abiertas.

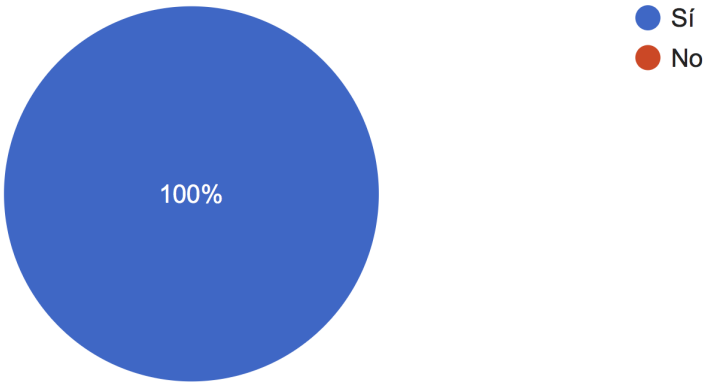
Evaluación de los encuestadores

Los encuestadores respondieron una evaluación acerca de la experiencia con las herramientas utilizadas para el relevamiento de datos, a continuación presentamos el resultado de la evaluación. Presentamos gráficos con frecuencias y proporciones para las preguntas con respuestas con valores cerrados, y una lista de las respuestas más relevantes para las preguntas abiertas.

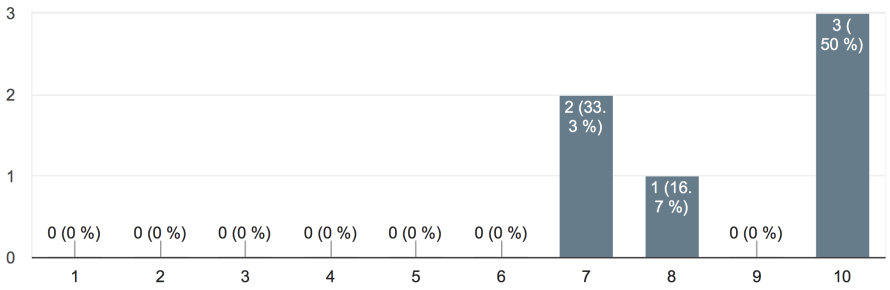
1. ¿Cómo calificarías la experiencia con el software para recolección de datos ODK?



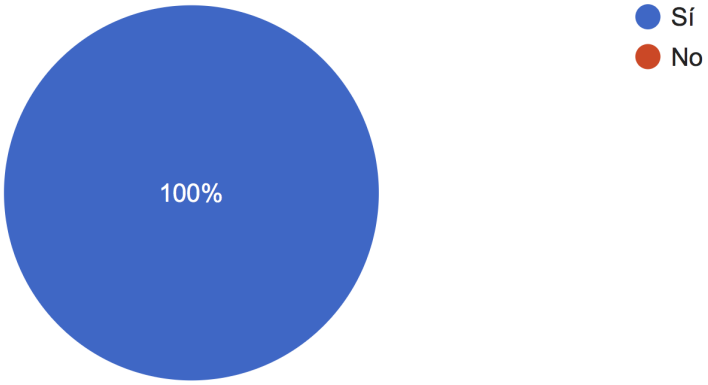
2. ¿Te pareció más fácil recolectar datos utilizando el software que utilizando un formulario de papel?



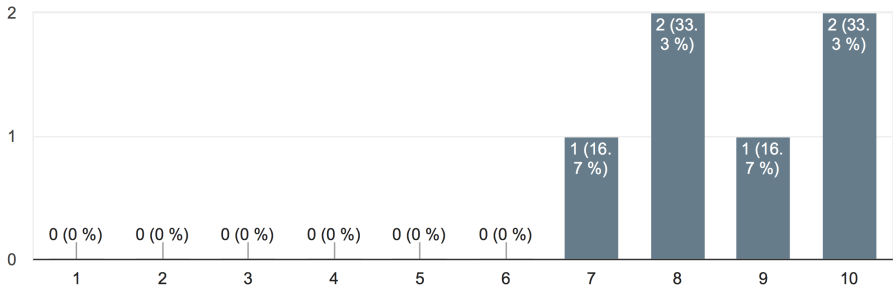
3. ¿En la escala del 1 al 10 qué tan fácil te pareció utilizar el software?



4. ¿Te pareció más rápido recolectar datos utilizando el software que utilizando un formulario de papel?



5. En la escala del 1 al 10 ¿qué tan rápido te pareció utilizar el software?



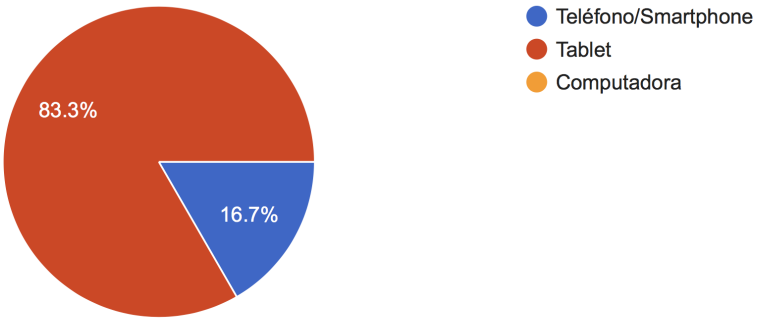
6. ¿Qué características del software de recolección de datos ODK Collect te gustaron?

   - La facilidad para la recolección de datos
   - GPS, anexo de fotos
   - Su practicidad, se podían recoger las informaciones necesarias de manera ágil y simple.
   - El hecho de no tener que manejar papeles para realizar las entrevistas y poder tener en una sola máquina todas las herramientas para recolectar los datos, no hacia falta un cúmulo de papeles para cada entrevista.

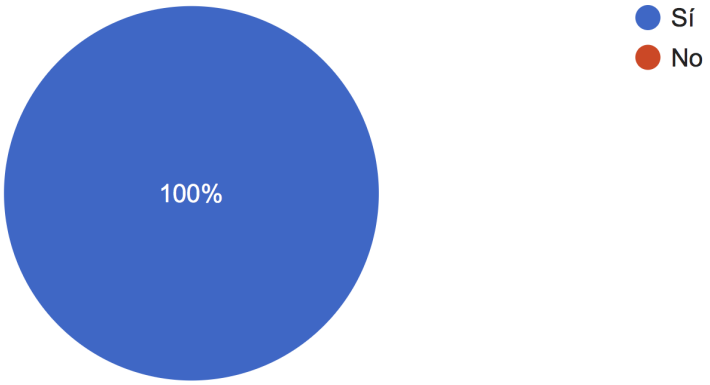
7. ¿Qué características del software de recolección de datos ODK Collect no te gustaron o te parecieron malas?

   - El error al envío de los datos.
   - La de departamentos y nombres de localidades, debería existir más discriminación.
   - Me pareció encontrar cuestiones referente al cuestionario de manera reiterada. Teniendo en cuenta que fueron 4 encuestas (diferentes) para cada hogar.
   - Que no se podía pasar a otra pregunta.
   - Teníamos cuatro planillas que llenar por cada entrevista y podíamos cometer el error de utilizar mas de una vez una planilla porque no tenia un mecanismo que bloquee o impida que se vuelva a utilizar en la misma entrevista, también que solo podíamos sacar una foto por entrevista, si sacábamos otra se borraba la foto anterior y en muchos casos una foto no era suficiente.

8. ¿En qué dispositivo te parecería más cómodo completar la encuesta?



9. ¿Usarías este software de recolección de datos para futuros casos de uso?



A partir de la evaluación de los encuestadores se pudo notar que la herramienta facilitaba la recolección de datos, destacando además sus características para la captura de multimedia. Como puntos en contra encontramos varios aspectos sobre el diseño de la aplicación y de la encuesta, la mayoría relacionadas con las opciones para los nombres de los departamentos y localidades en donde la selección múltiple sería mejor opción de formato de respuesta que la entrada de texto, también se resalta la repetitividad de las preguntas en las encuestas, esto podría optimizarse haciendo un rediseño de la encuesta de forma a minimizar la cantidad de preguntas a responder.

En cuanto al diseño de la aplicación, si bien la misma es sencilla e intuitiva, es posible que algunos usuarios no pudieran entender todas las funcionalidades con las que cuenta, en este caso, la solución sería una capacitación más completa del uso de la aplicación.

5.3.2. Análisis de datos

Se observó que la herramienta Open Data Kit provee herramientas de visualización de resultados en donde, dependiendo del caso, podrían llevar a alguna conclusión observando la distribución de los datos haciendo uso de los gráficos y mapas.

Con respecto al proceso de aprendizaje de máquina, para el caso de estudio de la encuesta "Cuestionario Domiciliar" las métricas de evaluación para el modelado del clasificador arrojaron resultados satisfactorios, lo que aporta confiabilidad a la hora de utilizarlo como predictor de las capturas de triatominos en el domicilio. Además el algoritmo utilizado para realizar la clasificación da como resultado un árbol de decisión, el mismo además puede ser utilizado para la extracción de conocimiento, que puede ser interpretado fácilmente por el usuario de la herramienta.

Basándonos en el árbol de decisión obtenido en la sección 5.11 para deducir alguna información útil respecto a la captura de vinchucas en el domicilio podemos concluir lo siguiente:

- Las viviendas que cuentan con energía eléctrica no arrojan una captura positiva.
- Los tipos de pared de la vivienda que favorecen la presencia de vinchucas son:
  - Pared francesa
  - Tronco
  - Ladrillo revocado
- Los tipos de techo de la vivienda que favorecen de presencia de vinchucas son:
  - Zinc
  - Tejas
  - Paja

Las conclusiones obtenidas del árbol de decisión coinciden con otros estudios sobre la prevalencia de triatominos en la vivienda. A continuación hacemos una comparación e interpretación de los factores que arrojan una captura positiva en el domicilio según el árbol obtenido.

En la raíz del árbol se infiere que las viviendas con energía eléctrica arrojan una captura negativa, esto podría deberse a que a las vinchucas no les agrada la luz [CCM02]. El hecho de tener energía eléctrica hace que las casas se iluminen en las noches y no arrojen capturas positivas. Además, el factor de no poseer energía eléctrica está asociado a la prevalencia de la enfermedad [HV15].

El siguiente nivel del árbol muestra al tipo de pared como determinante para la presencia de vinchucas. Se evidencia que las viviendas con construcciones precarias son preferidas por las vinchucas, además, las viviendas tipo "rancho" (con paredes de adobe, sin revoque) favorecen la presencia de las mismas, ya que los triatominos se alojan en las grietas de las paredes [CCM02]. El árbol de decisión indica que los tipos de pared más precarios tales como pared francesa o tronco son los factores que llevan a una captura positiva en la vivienda. Cabe destacar que el tipo de pared "ladrillo

revocado" no está asociado a una causa primaria de presencia de triatominos, pero existen evidencias que otros factores pueden influir a que este resultado esté presente. En el trabajo de [Arr+14] se evidencia que el ladrillo revocado y la presencia de animales domésticos muestran una correlación con la presencia de vinchucas, lo cual nos lleva a concluir que podrían existir otros factores que influencien indirectamente a esta variable, dichos factores podrían ser por ejemplo la higiene de la vivienda, la educación de los habitantes de la misma, etc, información que no fue recogida en la encuesta analizada.

Como último nivel del árbol encontramos el tipo de techo de la vivienda, destacando los tipos de techo con zinc, paja y tejas como aquellos factores que indican la presencia de vinchucas, este nodo se desprende del tipo de pared "tronco", lo cual nos indica que hay que tener en cuenta el tipo de techo. Como mencionamos anteriormente, las construcciones precarias son las más propensas a alojar vinchucas, tales como techos de paja, caña o barro [CCM02]. Si bien el tipo de techo con zinc y tejas no califican necesariamente como precario, nuevamente nos referimos a otros factores influyentes que quedaron fuera del alcance del estudio.

Capítulo 6

Conclusión y trabajos futuros

6.1. Conclusiones

En este trabajo presentamos una evaluación de herramientas de código abierto para la recolección de datos en campo utilizando teléfonos inteligentes y tabletas, con el fin de reemplazar formularios en papel, los cuales siguen siendo utilizados en la actualidad. Seleccionamos a la suite Open Data Kit (ODK) como herramienta a utilizar para el relevamiento de datos.

Implementamos las aplicaciones y herramientas de servidor de ODK para abordar el problema de la recolección de datos sobre infestación intra y peridomiciliar con triatominos vectores de la enfermedad de chagas en comunidades indígenas habitantes del Chaco Paraguayo, ejecutada por el Centro para el Desarrollo de la Investigación Científica (CEDIC). Se determinó que la herramienta seleccionada cumplió con los requerimientos de confiabilidad, almacenamiento y transmisión de los datos recogidos durante la encuesta, resaltando la capacidad de enriquecimiento de la información mediante datos de localización y multimedia de los formularios electrónicos.

En la segunda parte del trabajo abordamos la creación de una API para la realización de análisis de los datos recogidos, en la cual se apunta a la extracción de conocimiento y creación de modelos de predicción a partir de los datos relevados de las encuestas. Se hace uso de algoritmos y técnicas de aprendizaje de máquina para la implementación de clasificadores que puedan ser fácilmente entendibles para un usuario final. Los datos relevados de la encuesta sobre infestación de triatominos se utilizaron para la creación de un clasificador para predecir la presencia de vinchucas en hogares, el cual además fue utilizado para la visualización de factores que favorecen la infestación en la vivienda. Estos resultados luego fueron contrastados con estudios realizados sobre los factores principales, obteniendo resultados similares.

Concluimos que el conjunto de herramientas presentadas tiene un gran potencial para su aplicación en diversos ámbitos, en especial aquellos de rigor científico, recalcando el sector de salud. Por ejemplo, el mismo podría aplicarse para el relevamiento de datos de pacientes con patologías diversas. Además del valor de la digitalización

inmediata de los datos recabados se podrían extraer conclusiones de factores para dicha patología, e inclusive, una vez se ha creado un modelo de predicción, determinar si un paciente es propenso a desarrollar una enfermedad.

Este trabajo de grado produjo como resultados los siguientes ítems:

- Una herramienta validada para la recolección de datos utilizando formularios electrónicos.
- La agilización y mejoramiento de los tiempos de los procesos actuales de recolección utilizando un caso de estudio.
- Una documentación del proceso de implementación de recolección de datos.
- Una API para la creación de modelos de clasificación y extracción de conocimiento.
- La integración de estas herramientas open source de modo a validarlas en un caso de estudio.
- En conjunto, una plataforma open source genérica e integrada de recolección y análisis predictivo de datos, de bajo costo y de aplicación multidisciplinaria.

6.2. Trabajos futuros

En función a las conclusiones obtenidas presentamos a continuación una serie de propuestas que darían continuidad al trabajo:

- Desarrollar una herramienta de visualización de datos estadísticos con mayor variedad de gráficos basados en los datos recogidos de las encuestas.
- Agregar otros clasificadores de manera a seleccionar el de mejor desempeño para un conjunto de datos en particular. Estos clasificadores se podrían aplicar también a multimedia.
- Diseñar y desarrollar una interfaz gráfica para el módulo de análisis predictivo de datos, de modo a facilitar el uso a usuarios finales.
- Integrar los datos recabados con sistemas de georeferenciamiento para lograr un mayor entendimiento de los datos a través de mapas predictivos.
- Desarrollar una interfaz para la edición de datos recogidos desde el lado servidor.
- Agregar la posibilidad de cargar conjunto de datos de encuestas anteriores, de manera a utilizar estos datos como prueba del modelo de clasificación creado.

Apéndice A

Algoritmo de los K vecinos más cercanos (KNN)

Este algoritmo se utiliza para determinar los K vecinos más cercanos (K-nearest neighbors) a una tupla en particular (denominada tupla objetivo) dentro de un espacio n-dimensional, en donde n representa la cantidad de atributos de la tupla. La cercanía a una tupla se determina utilizando un criterio de distancia específico, esta medida podría ser la distancia euclidiana, la distancia de manhattan, u otra.

KNN utilizado por SMOTE (algoritmo 2) con el criterio de distancia euclidiana para determinar los K vecinos que son candidatos a ser seleccionados para crear la nueva instancia sintética.

La distancia euclidiana entre dos puntos o tuplas, sean $X_1 = (x_{11}, x_{12}, \ldots, x_{1n})$ y $X_2 = (x_{21}, x_{22}, \ldots, x_{2n})$, es

$$dist\,(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2} \tag{A.1}$$

En otras palabras, para cada atributo numérico, se toma la diferencia entre los valores correspondientes de un atributo en la tupla X_1 y la tupla X_2, se eleva al cuadrado esta diferencia y se acumula. Al final, se halla la raíz cuadrada del total resultante de la acumulación de distancias [HPK11].

La discusión anterior asume que los atributos utilizados para describir la tupla son numéricos. Para atributos nominales, un método simple consiste en comparar el valor del atributo correspondiente en la tupla X_1 con el valor de la tupla X_2. Si ambos son idénticos, (por ejemplo, las tuplas X_1 y X_2 tienen el mismo color azul), entonces la diferencia de ambos se toma como 0. Si son diferentes (por ejemplo, X_1 es azul y X_2 es rojo), entonces la diferencia se toma como 1. Otros métodos podrían incorporar esquemás más sofitiscados para realizar la diferencia.

Una vez calculadas las distancias a todas las tuplas, se devuelvan las K tuplas más cercanas. El procedimiento se muestra en el algoritmo 3.

Algoritmo 3 Algoritmo básico de los K vecinos más cercanos (KNN) [KW09]

Entrada: D, el conjunto de datos; z, la tupla objetivo; K, la cantidad de vecinos a devolver

Salida: Los K vecinos más cercanos a z

Método:

1) **para cada** instancia y en D **hacer**
2) Calcular $dist(y, z)$, la distancia entre y y z
3) **fin para**
4) Seleccionar N, donde $N \subseteq D$, el conjunto de K vecinos más cercanos de z utilizando las distancias calculadas
5) Devolver N

El algoritmo puede ser extremadamente lento cuando se tienen muchas tuplas. Si D es un conjunto de entrenamiento y $K = 1$, entonces, $O(|D|)$ comparaciones son requeridas para una tupla objetivo en particular. Utilizando árboles de búsqueda, el número de comparaciones puede ser reducido a $O(log(|D|))$, con implementaciones paralelas se puede reducir el tiempo de ejecución a una constante, $|O(1)|$, que sea independiente de $|D|$ [HPK11].

Apéndice B

Documentación de la API de análisis

`GET /forms`

Obtiene la lista de todos los formularios registrados en el sistema

Respuesta

`uri`	Identificador único del formulario
`dataModelUri`	Identificador del modelo del formulario. El modelo contiene la definición de la estructura de datos que aloja los datos recolectados en las encuestas.
`name`	Nombre del formulario
`author`	Nombre del usuario que creó el formulario
`creationDate`	Fecha de creación del formulario en formato Unix

Ejemplos

Petición

```
GET /forms
```

Respuesta

```
HTTP/1.1 200 Ok
[
  {
    "uri": "md5:f5d9e0d5aaad56b8dc330776c32832c5",
    "dataModelUri": "uuid:289ae1b7-a839-4111-a7e7-4ab75599a10a",
    "name": "Encuesta de hogares",
    "author": "user",
    "creationDate": 1441767280134
  },
  {...}
]
```

`GET /form/{uri}`

Obtiene la información de un formulio a través de su identificador.

Parámetros

`uri`	Identificador del formulario

Respuesta

`uri`	Identificador único del formulario
`dataModelUri`	Identificador del modelo del formulario. El modelo contiene la definición de la estructura de datos que aloja los datos recolectados en las encuestas.
`name`	Nombre del formulario
`author`	Nombre del usuario que creó el formulario
`creationDate`	Fecha de creación del formulario en formato Unix

Ejemplos

Petición

```
GET /forms/md5%3A6016c342e4bcbdca9e479d69f96f3652
```

Respuesta

```
HTTP/1.1 200 Ok
{
  "uri": "md5:6016c342e4bcbdca9e479d69f96f3652",
  "dataModelUri": "uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b",
  "name": "Cuestionario Domiciliar",
  "author": "user1",
  "creationDate": 1441767258313
}
```

`GET /form/{uri}/elements`

Obtiene información de los elementos de un formulario. Los elementos constituyen la estructura mediante la cual se definen las preguntas del formulario. Tipos de elementos:

- **STRING**: Texto
- **JRDATETIME**: Fecha/Hora
- **JRDATE**: Fecha
- **GROUP**: Grupo de elementos
- **INTEGER**: Números enteros
- **DECIMAL**: Números decimales
- **SELECTN**: Selección múltiple
- **GEOPOINT**: Datos de localización

Parámetros

`uri`	Identificador del formulario

Respuesta

`uri`	Identificador del elemento
`parentUri`	Identificador del elemento padre del elemento actual. Este campo sirve para determinar cuando un elemento forma parte de un grupo de elementos
`type`	Tipo de elemento
`name`	Nombre del elemento
`columnName`	Nombre de la columna que aloja los datos pertenecientes a este elemento en la base de datos.
`tableName`	Nombre de la tabla en donde se alojan los datos del elemento
`schemaName`	Nombre del esquema en donde reside la tabla de datos del elemento

Ejemplos

Petición

```
GET /forms/md5%3A6016c342e4bcbdca9e479d69f96f3652/elements
```

Respuesta

```
HTTP/1.1 200 Ok
[
  {
    "uri": "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000005)",
    "parentUri": "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000001)",
    "type": "STRING",
    "name": "deviceid",
    "columnName": "DEVICEID",
    "tableName": "CUE16LIAR_CORE",
    "schemaName": "odk_prod"
  },
  {...}
]
```

`GET /form/{uri}/data?urisToInclude={elementIds}`

Obtiene los datos recogidos para un formulario

Parámetros

`uri`	Identificador del formulario
`urisToInclude`	Lista de URIs de los elementos de los cuales obtener los datos. Si no se incluye este parámetro se obtienen los datos de todos los elementos

Respuesta

`total`	Número total de registros
`limit`	Número de registros en la página actual
`offset`	Número de página
`data`	Datos recogidos para el formulario. En cada elemento del array se tiene un objeto en donde se listan los nombres de los elementos con sus valores

Ejemplos

Petición

```
/forms/md5%3A0677e5a371f386a9c249a1e153f54018/data
?urisToInclude=elem%2Buuid%3A8ac309dc-c747-42df-8546-405a19a5d846(00000002)
%2Celem%2Buuid%3A8ac309dc-c747-42df-8546-405a19a5d846(00000004)
```

Respuesta

```
HTTP/1.1 200 Ok
{
  "total": 196,
  "limit": 10,
  "offset": 0,
  "data": [
    {
      "captura_peridomicilio": "captura_negativa",
      "nro_personas": 6,
      "antiguedad": 20
    }
  ]
}
```

`POST /forms/{uri}/analysis/predictor`

Crea un nuevo predictor (clasificador)

Parámetros

`uri`	Identificador del formulario
`uris`	Lista de id de los elementos a utilizar para la clasificación
`classUri`	Id del elemento que corresponde a la etiqueta de clase del conjunto de datos
`classifier`	Tipo del clasificador. Solo "J48" es soportado
`filter`	Nombre del filtro a aplicar antes de la clasificación. Por el momento solo se puede aplicar el filtro "SMOTE"
`filterParams`	Parámetros del filtro a utilizar. "SMOTE" recibe dos parámetros, `pct` indica el porcentaje de instancias minoritarias a crear como función del número actual de instancias, y `minorityClass` indica el valor de la clase a ser tomado como clase minoritaria.

Respuesta

numClasses	Número de clases
classNames	Nombre de las clases
classIsNominal	Indica si la clase es nominal
numInstances	Número de instancias
accuracy	Exactitud
errorRate	Tasa de error
correctlyClassifiedInstances	Número de instancias clasificadas correctamente
pctCorrectlyClassifiedInstances	Porcentaje de instancias clasificadas correctamente
unClassifiedInstances	Número de instancias no clasificadas
pctUnClassifiedInstances	Porcentaje de instancias no clasificadas
incorrectlyClassifiedInstances	Número de instancias clasificadas incorrectamente
pctIncorrectlyClassifiedInstances	Porcentaje de instancias clasificadas incorrectamente
detailedAccuracyByClass	Lista de objetos que contiene una evaluación por clase, cada elemento del array tiene los siguientes campos
sensitivity	Sensitividad
specificity	Especificidad
precision	Precisión
recall	Exhaustividad
fMeasure	Valor F
areaUnderROC	Área bajo la curva ROC
class	Nombre de la clase
confusionMatrix	Matriz de confusión
titleRow	Nombre de las filas de la matriz de confusión
classColumn	Nombre de las columnas de la matriz de confusión
matrix	Valores de la matriz de confusión

Ejemplos

Petición

```
POST /forms/md5%3A0677e5a371f386a9c249a1e153f54018/data/analysis/predictor
{
  "uris": [
    "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000022)",
    "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000008)",
    "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000016)"
  ],
  "classUri": "elem+uuid:1c5e87fb-ce77-46b5-aed6-747168c43b0b(00000030)",
  "classifier": "J48",
  "filter" : "SMOTE",
  "filterParams" : {
    "pct" : 100,
    "minorityClass" : "captura_positiva"
  }
```

```
  }
```

Respuesta

```
HTTP/1.1 201 Created
Location: /forms/md5%3A0677e5a371f386a9c249a1e153f54018/analysis/predictor/1
{
  "numClasses": 2,
  "classNames": [ "captura_negativa",
                  "captura_positiva" ],
  "confusionMatrix": {
    "titleRow": [ "a", "b" ],
    "classColumn": [ "a = captura_negativa",
                     "b = captura_positiva" ],
    "matrix": [ [ 138.0, 0.0 ],
                [ 10.0, 0.0 ] ]
  },
  "correctlyClassifiedInstances": 138.0,
  "pctCorrectlyClassifiedInstances": 93.24324324324324,
  "incorrectlyClassifiedInstances": 10.0,
  "pctIncorrectlyClassifiedInstances": 6.756756756756757,
  "unClassifiedInstances": 0.0,
  "pctUnClassifiedInstances": 0.0,
  "numInstances": 148.0,
  "accuracy": 0.9324324324324325,
  "errorRate": 0.06756756756756754,
  "detailedAccuracyByClass": [
    {
      "sensitivity": 1.0,
      "specificity": 0.9,
      "precision": 0.9324324324324325,
      "recall": 1.0,
      "fMeasure": 0.965034965034965,
      "areaUnderROC": 0.49075462268865566,
      "class": "captura_negativa"
    }, {
      "sensitivity": 0.0,
      "specificity": 0.0,
      "precision": 0.0,
      "recall": 0.0,
      "fMeasure": 0.0,
      "areaUnderROC": 0.4967741935483871,
      "class": "captura_positiva"
    }
  ]
}
```

`GET /form/{uri}/analysis/predictor/{id}`

Obtiene la evaluación de un predictor ya creado a partir de su identificador

Parámetros

`uri`	Identificador del formulario
`id`	Identificador del predictor

Respuesta

`id`	Identificador del predictor
`type`	Tipo, equivalente al algoritmo utilizado para construir el predictor
`evaluation`	Evaluación del predictor, contiene todos los campos a continuación.
`numClasses`	Número de clases
`classNames`	Nombre de las clases
`classIsNominal`	Indica si la clase es nominal
`numInstances`	Número de instancias
`accuracy`	Exactitud
`errorRate`	Tasa de error
`correctlyClassifiedInstances`	Número de instancias clasificadas correctamente
`pctCorrectlyClassifiedInstances`	Porcentaje de instancias clasificadas correctamente
`unClassifiedInstances`	Número de instancias no clasificadas
`pctUnClassifiedInstances`	Porcentaje de instancias no clasificadas
`incorrectlyClassifiedInstances`	Número de instancias clasificadas incorrectamente
`pctIncorrectlyClassifiedInstances`	Porcentaje de instancias clasificadas incorrectamente
`detailedAccuracyByClass`	Lista de objetos que contiene una evaluación por clase, cada elemento del array tiene los siguientes campos
`sensitivity`	Sensitividad
`specificity`	Especificidad
`precision`	Precisión
`recall`	Exhaustividad
`fMeasure`	Valor F
`areaUnderROC`	Área bajo la curva ROC
`class`	Nombre de la clase
`confusionMatrix`	Matriz de confusión
`titleRow`	Nombre de las filas de la matriz de confusión
`classColumn`	Nombre de las columnas de la matriz de confusión
`matrix`	Valores de la matriz de confusión

Ejemplos

Petición

```
GET /forms/md5%3A0677e5a371f386a9c249a1e153f54018/data/analysis/predictor/0
```

Respuesta

```
HTTP/1.1 200 Ok
{
```

```
  "id": 0,
  "type": "J48",
  "evaluation": {
    "numClasses": 2,
    "classNames": [ "captura_negativa",
                    "captura_positiva" ],
    "confusionMatrix": {
      "titleRow": [ "a", "b" ],
      "classColumn": [ "a = captura_negativa",
                       "b = captura_positiva" ],
      "matrix": [ [ 138.0, 0.0 ],
                  [ 10.0, 0.0 ] ]
    },
    "correctlyClassifiedInstances": 138.0,
    "pctCorrectlyClassifiedInstances": 93.24324324324324,
    "incorrectlyClassifiedInstances": 10.0,
    "pctIncorrectlyClassifiedInstances": 6.756756756756757,
    "unClassifiedInstances": 0.0,
    "pctUnClassifiedInstances": 0.0,
    "numInstances": 148.0,
    "accuracy": 0.9324324324324325,
    "errorRate": 0.06756756756756754,
    "detailedAccuracyByClass": [
      {
        "sensitivity": 1.0,
        "specificity": 0.9,
        "precision": 0.9324324324324325,
        "recall": 1.0,
        "fMeasure": 0.965034965034965,
        "areaUnderROC": 0.49075462268865566,
        "class": "captura_negativa"
      }, {
        "sensitivity": 0.0,
        "specificity": 0.0,
        "precision": 0.0,
        "recall": 0.0,
        "fMeasure": 0.0,
        "areaUnderROC": 0.4967741935483871,
        "class": "captura_positiva"
      }
    ]
  }
}
```


GET /form/{uri}/analysis/predictor/{id}/representation

Obtiene la representación gráfica del predictor si es que lo posee. Actualmente retorna una representación del árbol de decisión para predictores de tipo J48.

Parámetros

`uri`	Identificador del formulario
`id`	Identificador del predictor

Respuesta

`representation`	Representación gráfica del predictor. Para el predictor de tipo J48 se retorna el árbol de decisión en formato "dot" (Graphviz)

Ejemplos

Petición

```
GET /forms/md5%3A0677e5a371f386a9c249a1e153f54018/analysis
/predictor/0/representation
```

Respuesta

```
HTTP/1.1 200 Ok
{
  "representation":
    "digraph J48Tree {N0 [label=\"'captura_negativa (148.0/10.0)'\"
    shape=box style=filled ]}"
}
```

POST /form/{uri}/analysis/predictor/{id}/prediction

Realiza un predicción utilizando un predictor

Parámetros

`uri`	Identificador del formulario
`id`	Identificador del predictor
`payload`	Se requiere un objeto en donde se envian los valores de los elementos por uri, en formato clave/valor. Referirse al ejemplo

Respuesta

`classification`	Valor resultante de la clasificación

Ejemplos

Petición

```
POST /forms/md5%3A0677e5a371f386a9c249a1e153f54018/analysis/predictor/0/prediction
{
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000007)": "Campo Largo",
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000008)": 2,
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000014)": 2,
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000015)": 3,
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000016)": 2,
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000018)": "Tronco",
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000019)": "Paja",
  "elem+uuid:9e709eb0-3491-4815-871c-b1bd54fa3d33(00000020)": "Tierra"
}
```

Respuesta

```
HTTP/1.1 200 Ok
{
  "classification": "captura_negativa"
}
```

Bibliografía

[Rec14] D Recalde. «Calidad del llenado de la Planilla Semanal de Notificación Obligatoria». En: *Revista Paraguaya de Epidemiología* 2.1 (2014), págs. 27-40.

[Col12] ABC Color. *Comienza hoy el Censo Nacional de Población 2012*. 2012. URL: http://www.abc.com.py/edicion-impresa/economia/arranca-hoy-el-censo-nacional-2012-que-se-extendera-por-unas-6-semanas-464280.html (visitado 15-10-2016).

[Pob12] Fondo de Población de las Naciones Unidas. «Paraguay frente al Censo 2012». En: *Joparé Paraguay* 47 (ene. de 2012).

[Cona] Dirección Nacional de Contrataciones Públicas. *Contrato de la Licitación 309599 - Software*. URL: https://www.contrataciones.gov.py/licitaciones/adjudicacion/contrato/309599-sodep-s-a-5.html#itemsLote (visitado 18-12-2017).

[Koe] Emily C. Koenig. *Market Researchers Adapt to Smartphones*. URL: https://www.surveysampling.com/blog/market-researchers-adapt-to-smartphones/ (visitado 29-11-2017).

[Sal12] Ministerio de Salud Pública y Bienestar Social. *Primera Encuesta Nacional de Factores de Riesgo de Enfermedades no Transmisibles*. 2012.

[Conb] Dirección Nacional de Contrataciones Públicas. *Contrato de la Licitación 301664 - Licencia de Software para el Programa TENONDERA*. URL: https://www.contrataciones.gov.py/licitaciones/adjudicacion/contrato/301664-sodep-s-a-1.html#itemsLote (visitado 18-12-2017).

[SEN] SENEPA. *Programas del SENEPA*. URL: http://programassenepa.blogspot.com/p/chagas.html (visitado 15-10-2016).

[Des11] Centro para el Desarrollo de la Investigación Científica. *Cómo hacerles caer en la trampa*. 2011. URL: http://www.cedicpy.com/blog/2011/11/02/como-hacerles-caer-en-la-trampa.

[Arr+14] Cristina Arrom y col. «Comportamientos que favorecen la dinámica de reinfestación de Triatoma infestans del Chaco paraguayo». En: *Memorias del Instituto de Investigaciones en Ciencias de la Salud* 11.2 (2014), págs. 7-15.

[Sal11] Organización Panamericana de la Salud/Organización Mundial de la Salud. «Implementación de un Sistema de Vigilancia para el control de de la Enfermedad de Chagas con Participación comunitaria en el Parguay 2002 - 2010». En: (2011), pág. 56.

[Mag] Magpi. *Paper Data Collection: An Environmental Disaster*. URL: `http://home.magpi.com/paper-data-collection-environmental-disaster` (visitado 15-10-2016).

[Med+15] Araya Abrha Medhanyie y col. «Mobile health data collection at primary health care in Ethiopia: a feasible challenge». En: *Journal of clinical epidemiology* 68.1 (2015), págs. 80-86.

[Nju+14] Henry N Njuguna y col. «A comparison of smartphones to paper-based questionnaires for routine influenza sentinel surveillance, Kenya, 2011–2012». En: *BMC medical informatics and decision making* 14.1 (2014), pág. 1.

[Zha+12] Shuyi Zhang y col. «Smartphone versus pen-and-paper data collection of infant feeding practices in rural China». En: *Journal of medical Internet research* 14.5 (2012), e119.

[Kita] Open Data Kit. *About Open Data Kit*. URL: `https://opendatakit.org/about/` (visitado 16-10-2016).

[W3C] W3C. *XForms 1.1*. URL: `http://www.w3.org/TR/xforms` (visitado 16-10-2016).

[Enk] Enketo. *Introduction to OpenRosa*. URL: `https://enketo.org/openrosa` (visitado 16-10-2016).

[Har+10] Carl Hartung y col. «Open data kit: tools to build information services for developing regions». En: *Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development*. ACM. 2010, pág. 18.

[Kitb] Open Data Kit. *2.0 Tool Suite*. URL: `https://opendatakit.org/use/2_0_tools/` (visitado 16-10-2016).

[Kitc] Open Data Kit. *The future of Open Data Kit*. URL: `https://opendatakit.org/2016/08/the-future-of-open-data-kit/` (visitado 16-10-2016).

[Pap] Enketo Smart Paper. *Enketo Smart Paper - Next Generation Webforms*. URL: `https://enketo.org` (visitado 16-10-2016).

[For] Formhub. *Formhub*. URL: `http://formhub.org` (visitado 16-10-2016).

[Ini] Harvard Humanitarian Initiative. *KoBoToolbox | Data Collection Tools for Challenging Environments*. URL: `http://www.kobotoolbox.org` (visitado 16-10-2016).

[Tom+15] Daniel Tom-Aba y col. «Innovative Technological Approach to Ebola Virus Disease Outbreak Response in Nigeria Using the Open Data Kit and Form Hub Technology». En: *PloS one* 10.6 (2015), e0131000.

[Pud16] Nirab Pudasaini. *Open source and open data's role in Nepal earthquake relief*. 2016. URL: `https://opensource.com/life/16/6/open-source-open-data-nepal-earthquake` (visitado 16-10-2016).

[Bru15] Waylon Brunette. *Reducing errors and delays in collecting millions of World Food Programme data points*. 2015. URL: `https://opendatakit.org/2015/03/reducing-errors-and-delays-in-collecting-millions-of-world-food-programme-data-points/` (visitado 16-10-2016).

[Infa] Epi Info. *Epi Info™ | CDC*. URL: `http://www.cdc.gov/epiinfo/index.html` (visitado 24-10-2016).

[Infb] Epi Info. *Epi Info™ - Community Edition - Home*. URL: `https://epiinfo.codeplex.com/` (visitado 14-11-2016).

[MFO02] Alicia Milano, María Francista y Elena Breatriz Oscherov. «Contaminación por parásitos caninos de importancia zoonótica en playas de la ciudad de Corrientes, Argentina». En: *Parasitología latinoamericana* 57.3-4 (2002), págs. 119-123.

[Epia] EpiCollect. *EpiCollect*. URL: `http://www.epicollect.net` (visitado 16-10-2016).

[Epib] EpiCollect. *EpiCollect+*. URL: `http://www.epicollect.net/plus_Instructions/developers/default.html` (visitado 16-10-2016).

[Aan+09] David M Aanensen y col. «EpiCollect: linking smartphones to web applications for epidemiology, ecology and community data collection». En: *PloS one* 4.9 (2009), e6968.

[HPK11] Jiawei Han, Jian Pei y Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[ASR15] Aida Ali, Siti Mariyam Shamsuddin y Anca L Ralescu. «Classification with class imbalance problem: a review». En: *Int. J. Advance Soft Compu. Appl* 7.3 (2015).

[Tan+06] Pang-Ning Tan y col. *Introduction to data mining*. Pearson Education India, 2006.

[Bei06] Steven M Beitzel. «On understanding and classifying web queries». Tesis doct. Citeseer, 2006.

[WF05] Ian H Witten y Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[Jar14] Néstor Concepción Jara Landolffi. «Análisis de los factores asociados al parto pretérmino en la Cátedra Clínica Gineco-Obstétrica de la Facultad de Ciencias Médicas de la Universidad Nacional de Asunción». Tesis de mtría. Universidad Nacional de Asunción, 2014.

[SWK09] Yanmin Sun, Andrew KC Wong y Mohamed S Kamel. «Classification of imbalanced data: A review». En: *International Journal of Pattern Recognition and Artificial Intelligence* 23.04 (2009), págs. 687-719.

[LD13] Rushi Longadge y Snehalata Dongre. «Class imbalance problem in data mining review». En: *arXiv preprint arXiv:1305.1707* (2013).

[Cha+02] Nitesh V Chawla y col. «SMOTE: synthetic minority over-sampling technique». En: *Journal of artificial intelligence research* 16 (2002), págs. 321-357.

[Chi13] Eric Chio. *Class Imbalance Problem*. 2013. URL: `http://www.chioka.in/class-imbalance-problem/`.

[Gar] Gartner. *Gartner Says Five of Top 10 Worldwide Mobile Phone Vendors Increased Sales in Second Quarter of 2016*. URL: `http://www.gartner.com/newsroom/id/3415117` (visitado 01-12-2016).

[Dev] Android Developers. *Paneles de control | Android Developers*. URL: `https://developer.android.com/about/dashboards/index.html#Platform` (visitado 06-12-2016).

[ITU] International Telecommunication Union (ITU). *X.667 : Tecnología de la información - Interconexión de sistemas abiertos - Procedimientos para el funcionamiento de autoridades de registro OSI: Generación y registro de identificadores únicos universales y su utilización como componentes de identificador de objetos ASN.1*. URL: `http://www.itu.int/rec/T-REC-X.667-200409-S/es` (visitado 13-01-2017).

[Kitd] Open Data Kit. *Aggregate Database Structure*. URL: `https://github.com/opendatakit/opendatakit/wiki/Aggregate-Database-Structure` (visitado 15-12-2016).

[Mac] University of Waikato Machine Learning Group. *Weka 3: Data Mining Software in Java*. URL: `http://www.cs.waikato.ac.nz/ml/weka` (visitado 04-01-2017).

[KW09] Vipin Kumar y Xindong Wu. *The top ten algorithms in data mining*. CRC Press, 2009.

[CCM02] L Crocco, S Catalá y M Martínez. «Enfermedad de Chagas: módulo de actualización». En: *Córdoba: Editorial Universitas* (2002).

[HV15] Pilar Hernandez y Montserrat de Villasante Fuentes. «Novedades en el diagnóstico de la enfermedad de Chagas». En: 11 (mar. de 2015), págs. 141-145.