DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: PREDICCIÓN DE ALERTAS DE INCIDENTES PARA UN SERVICIO INTEGRADO DE SEGURIDAD MEDIANTE APRENDIZAJE DE MÁQUINAS

AUTOR: PATIÑO VELA, CHRISTOPHER ALEXANDER

DIECTOR: ING. CARRERA ERAZO, ENRIQUE VINICIO

SANGOLQUÍ

2018

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, "***PREDICCIÓN DE ALERTAS DE INCIDENTES PARA UN SERVICIO INTEGRADO DE SEGURIDAD MEDIANTE APRENDIZAJE DE MÁQUINAS***" realizado por el señor ***CHRISTOPHER ALEXANDER PATIÑO VELA***, el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 7 de marzo del 2018

ENRIQUE VINICIO CARRERA ERAZO

C.C. 1708792104

DIRECTOR

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, ***CHRISTOPHER ALEXANDER PATIÑO VELA***, declaro que el contenido, ideas y criterios del trabajo de titulación: "***PREDICCIÓN DE ALERTAS DE INCIDENTES PARA UN SERVICIO INTEGRADO DE SEGURIDAD MEDIANTE APRENDIZAJE DE MÁQUINAS"*** es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 7 de marzo del 2018

CHRISTOPHER ALEXANDER PATIÑO VELA

C.C. 1718161688

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, ***CHRISTOPHER ALEXANDER PATIÑO VELA***, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación ***"PREDICCIÓN DE ALERTAS DE INCIDENTES PARA UN SERVICIO INTEGRADO DE SEGURIDAD MEDIANTE APRENDIZAJE DE MÁQUINAS"*** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 7 de marzo del 2018

CHRISTOPHER ALEXANDER PATIÑO VELA

C.C. 1718161688

DEDICATORIA

Dedico este proyecto de grado a mis padres por ser quienes hicieron de mi la persona que soy ahora, por el cariño que me brindaron y las enseñanzas que me han permitido crecer como persona y profesionalmente, a mi hermana quien ha sido mi compañera y mi amiga desde siempre y que de seguro estará para mí cuando la necesite y yo para ella, a mi abuelita quien fue mi segunda madre y que desearía poder tener mi lado, y al resto de mi familia porque cada uno ha sido el ejemplo oportuno que necesité en cada etapa de mi vida.

AGRADECIMIENTO

Agradezco primero a Dios por permitirme haber llegado hasta este punto de mi vida, a mis padres por el cariño, por ser mi guía y apoyarme con todo lo necesario, a mi hermana por ser mi amiga y apoyo, a mi abuelita por darme apoyo sentimental y material, a mi tutor de proyecto de grado por guiarme, apoyarme y tener paciencia, a mis amigos por la ayuda logística para poder terminar el proyecto, y al resto de mis profesores por darme el conocimiento necesario para crecer como profesional.

ÍNDICE DE CONTENIDO

CERTIFICACIÓN i

AUTORÍA DE RESPONSABILIDAD ii

AUTORIZACIÓN iii

DEDICATORIA iv

AGRADECIMIENTO v

ÍNDICE DE CONTENIDO vi

RESUMEN ii

ABSTRACT iii

CAPÍTULO I 1

1. INTRODUCCIÓN 1

1.1. Antecedentes 1

1.2. Justificación e Importancia 3

1.3. Alcance del Proyecto 4

1.4. Objetivos 6

1.4.1. Objetivo General 6

1.4.2. Objetivos Específicos 6

CAPÍTULO II 8

2. FUNDAMENTO TEÓRICO 8

2.1. Aprendizaje de máquinas. 8

2.1.1. Tipos de aprendizaje. 8

2.1.2. SVM. 10

2.1.2.1. SVM para clasificación binaria 10

2.1.2.2. SVM para regresión 17

2.1.3. Evaluación de máquinas de aprendizaje. 20

2.2. Matlab y SVM. 21

2.2.1. Función *fitrsvm* y *fitcsvm* 22

2.2.2. Función *fitrlinear* y *fitclinear* 23

2.2.3. Función predict 24

CAPÍTULO III 25

3. IMPLEMENTACIÓN DEL MODELO DE PREDICCIÓN DE INCIDENTES 25

3.1. Extracción de datos para la creación de nuevos registros. 25

3.1.1. Algoritmo para la extracción de datos. 27

3.1.2. Estructuración de registros a partir de los datos extraídos. 28

3.2. Preparación de los datos para el ingreso en el aprendizaje de máquina. 31

3.2.1. Procesamiento de datos de entrada del modelo SVM. 31

3.2.1.1. Conversión a datos numéricos. 31

3.2.1.2. Cuantificación de los valores de respuesta 32

3.2.1.3. Escalado exponencial de los valores de respuesta.....33

3.2.1.4. Codificación binaria de los valores de respuesta 35

3.2.2. Estructuración de una nueva base de datos del año 2014 procesada.....35

3.3. Implementación del modelo SVM para la predicción del número de incidentes. 38

3.4. Implementación del modelo SVM para la predicción del tipo de incidentes. .40

3.5. Predicción de datos usando las máquinas generadas. 41

3.6. Procesamiento de datos de salida. 41

3.7. Creación de una interfaz gráfica en Matlab para el ingreso de datos y visualización de resultados. 43

CAPÍTULO VI.....46

4. RESULTADOS.....46

4.1. Evaluación del Kernel y selección de hiperparámetros óptimos 47

4.2. Evaluación de las máquinas creadas del número de incidentes.....50

4.3. Evaluación de las máquinas creadas según los tipos de incidentes.....55

CAPÍTULO VII.....66

5. CONCLUSIONES Y RECOMENDACIONES 66

5.1. Conclusiones.....66

5.2. Recomendaciones y trabajos futuros. 68

REFERENCIAS..70

ÍNDICE DE TABLAS

Tabla 1 *Características de la base de datos original*.....26

Tabla 2 *Ejemplo del formato de la nueva base de datos de número de incidentes* 29

Tabla 3 *Código para cada tipo de incidente*.....30

Tabla 4 *Ejemplo del formato de la nueva base de datos de tipo de incidentes*.....30

Tabla 5 *Ejemplo de la conversión numérica de las nuevas bases de datos* 32

Tabla 6 *Ejemplo de la cuantificación uniforme de la nueva base de datos*.....36

Tabla 7 *Ejemplo de la cuantificación y escalado exponencial de la nueva base de datos*.....37

Tabla 8 *Ejemplo de la cuantificación y codificación de la nueva base de datos*.....37

Tabla 9 *Ejemplo del formato de la nueva base de datos de tipo de incidentes con la característica complementaria*. 38

Tabla 10 *Error con datos originales y normalizados* 48

Tabla 11 *Pruebas para la selección de los hiperparámetros óptimos* 49

Tabla 12 *Hiperparámetros óptimos kernel gaussiano (número de incidentes)* 51

Tabla 13 *Hiperparámetros óptimos kernel polinómico (número de incidentes)*.....51

Tabla 14 *Hiperparámetros óptimos kernel lineal (número de incidentes)* 52

Tabla 15 *Hiperparámetros óptimos kernel gaussiano (tipo de incidentes)*.....56

Tabla 16 *Hiperparámetros óptimos kernel polinómico (tipo de incidentes)* 56

Tabla 17 *Hiperparámetros óptimos kernel lineal (tipo de incidentes)*.....57

Tabla 18 *Hiperparámetros óptimos kernel gaussiano (número de incidentes)* 60

Tabla 19 *Hiperparámetros óptimos kernel polinómico (número de incidentes)*.....60

Tabla 20 *Hiperparámetros óptimos kernel lineal (número de incidentes)* 61

Tabla 21 *Hiperparámetros óptimos kernel gaussiano (tipo de incidentes)* 61

Tabla 22 *Hiperparámetros óptimos kernel polinómico (tipo de incidentes)* 61

Tabla 23 *Hiperparámetros óptimos kernel lineal (tipo de incidentes)* 62

ÍNDICE DE FIGURAS

Figura 1 Ejemplos separables linealmente 12

Figura 2 Ejemplo cuasi-separable linealmente 14

Figura 3 SVM para regresión 18

Figura 4 Ejemplo cuantificación y escalado a 128 niveles 34

Figura 5 Interfaz gráfica 45

Figura 6 Respuesta cuantificada uniformemente (512 niveles-por muestra) 54

Figura 7 Respuesta cuantificada uniformemente (512 niveles-por respuesta) 54

Figura 8 Respuesta cuantificada uniformemente (512 niveles-error relativo por respuesta) 55

Figura 9 Respuesta numérica original, tipo de incidentes (por muestra) 58

Figura 10 Respuesta numérica original, tipo de incidentes (por respuesta) 58

Figura 11 Respuesta numérica original, tipo de incidentes (error relativo por respuesta) 59

Figura 12 Histograma de comparación de modelos SVM cuantificados y con escala exponencial. 63

RESUMEN

Un Servicio Integrado de Seguridad se encarga de atender las llamadas de emergencia de la ciudadanía para prestar la debida atención mediante la asignación de los recursos necesarios que requiera el incidente, a través de las instituciones de policía, bomberos, sanidad, militares, salud, etc.; por el hecho de que se debe atender la solicitud de una gran cantidad de emergencias no previstas, hay ocasiones en que no se tienen los recursos listos, resultando en consecuencias fatales. Para que el sistema mencionado pueda ser más eficiente en el aspecto de disponibilidad de recursos y por lo tanto en la atención de emergencias, se implementa un software capaz de predecir el número y el tipo de incidentes que puedan suscitarse en un distrito, mes, día de la semana y turno (horario) específicos, para lo cual se toma como ejemplo el Servicio Integrado de Seguridad ECU 911 y el centro zonal Quito específicamente. Dicho objetivo se realiza mediante la aplicación de aprendizaje de máquinas, exactamente las máquinas de vectores soporte (SVM) de clasificación y regresión, utilizando bases de datos de llamadas de emergencias del año 2014 proporcionada por la empresa. Después de realizar una serie de pruebas de diferentes modelos SVM con distintos métodos de tratamiento de datos y basándose en el mejor rendimiento de acuerdo al MAE y al error relativo, se elige el mejor modelo con respuestas cuantificadas a escala exponencial a 512 niveles y kernel gaussiano, con el cual se crea una interfaz gráfica amigable para manejo de cualquier usuario.

PALABRAS CLAVE:

- **EMERGENCIAS**
- **ECU 911**
- **APRENDIZAJE DE MÁQUINAS**
- **MÁQUINAS DE VECTORES SOPORTE**
- **REGRESIÓN**

ABSTRACT

An Integrated Security Service is responsible for responding to the emergency calls of the citizens to give due attention through the assignment of the necessary resources that required the situation or incident, by police and fire department, sanitation, military and health institutions, etc.; because they should request a large number of unforeseen emergencies, there are many times when resources are not ready, that results in fatal consequences. With the purpose of the mentioned system could be more efficient in the aspect of resource availability and therefore in emergency care, a software tool is implemented that can predict the number and type of incidents that may occur in a specific district, month, day of the week and time shift (schedule), for which we take as an example the Integrated Security Service ECU 911 and the Quito operations center specifically. This objective is achieved by the application of machine learning, specifically the classification and regression support vector machines (SVM), using databases of emergency calls of the 2014 provided by the company. After performing a serial of tests of different SVM models with different data processing methods and based on the best performance according to the MAE and Relative Error, the best model is chosen with exponential scale quantified responses to 512 levels and Gaussian kernel, with which a friendly graphical interface is created that can be handled by any user.

KEY WORDS:

- **EMERGENCY**
- **ECU 911**
- **MACHINE LEARNING**
- **SUPPORT VECTOR MACHINES**
- **REGRESSION**

CAPÍTULO I

1. INTRODUCCIÓN

1.1. Antecedentes

Para los antecedentes es necesario describir cómo funciona actualmente un sistema de Servicio Integrado de Seguridad como por ejemplo el ECU 911 en el Ecuador y la predicción de sistemas mediante el aprendizaje de SVM, como se hace a continuación.

1.1.1. Sistema del ECU 911

El Servicio Integrado de Seguridad (SIS) ECU 911 es la institución ecuatoriana encargada de responder a cualquier tipo de emergencia que sufra la población en cada rincón del país, a través de llamadas al número de emergencia 911, videovigilancia, botones de auxilio y alarmas comunitarias, las 24 horas del día, los 365 días al año, para así contribuir permanentemente a la consecución y mantenimiento de la seguridad integral ciudadana.

En la actualidad se trata de un servicio integrado en el que participan los diferentes organismos de atención a emergencias como son: Policía Nacional, Ministerio de Defensa Nacional, Cuerpo de Bomberos, Cruz Roja Ecuatoriana, Ministerio de Salud Pública, Instituto Ecuatoriano de Seguridad Social y Secretaría Nacional de Gestión de

Riesgos. Así reemplaza a todos los números de emergencia que fragmentaba la atención en distintas instituciones de respuesta (Servicio Integrado de Seguridad ECU 911, 2016).

Sin embargo, y por el hecho de que existen aproximadamente 1300 llamadas de emergencia diarias (ECU-911 Quito, 2013) y de las cuales un gran porcentaje son falsas, pero igual son atendidas, no siempre existen los recursos necesarios en el momento exacto para atender todas las emergencias suscitadas.

1.1.2. Predicción

La predicción se realiza para obtener información anticipada del comportamiento en el tiempo de diferentes tipos de sistemas, eventos inciertos o datos que son difíciles de conseguir para poder realizar estudios con ellos o simplemente saber el comportamiento de dichos sistemas. Para la predicción de series temporales, que se refieren a que sus estimaciones futuras dependen únicamente de los datos del pasado y no de variables independientes, se toman en cuenta tres métodos que son: el ARIMA (Modelo Autoregresivo Integrado de Promedio Móvil) que es estadístico, MLP (Perceptrón Multicapa) y SVM (Máquina de Vectores Soporte), siendo estos dos últimos métodos de aprendizaje de máquinas, de los cuales el SVM presenta mayor eficiencia en la predicción de series no lineales por su alta capacidad de generalización de acuerdo a la elección adecuada de sus parámetros y la utilización de la función núcleo gaussiana (Velásquez, Olaya, & Franco, 2010).

Las SVM para predicción son usadas en varios tipos de sistemas, así como la predicción de la temperatura atmosférica (clima) que puede llegar a valores de MSE

(Erros Cuadrático Medio) entre 7.07 y 7.56 comparado con un MLP con valores entre 8.07 y 10.2 (Radhika & Shashi, 2009); dentro de la biología la predicción de las estructuras de la proteína con una exactitud del 99% (Protein Structure Prediction Using Support Vector Machine, 2012), e incluso la predicción de sistemas tan complejos como la bolsa de valores por su dinamismo y que puede llegar a una exactitud del 74.4% (Shunrong, Haomiao, & Tongda, 2012).

1.2. Justificación e Importancia

De acuerdo a los antecedentes presentados, es necesario que los organismos aliados cuenten con los recursos humanos y materiales suficientes para cada emergencia en el instante en que se produzca, es decir el momento mismo en que la alerta sea confirmada por un servidor de la institución, logrando una respuesta rápida y asegurando el bienestar de la o las personas implicadas en la emergencia.

La única forma de asegurar los recursos necesarios para cada emergencia, en especial los recursos humanos, es sabiendo cuantas emergencias y de qué tipo se van a producir en un sector específico, en un determinado día y turno, ya que no es posible tener una cantidad indefinida de personal lista para actuar en todos los organismos aliados todo el tiempo y en todo lado.

Es posible construir un sistema en software que obtenga una predicción de estos datos basándose en los datos recompilados de años anteriores, logrando una cantidad aproximada y que es única para cada día del año, ya que cada día es diferente por depender del movimiento social que se produce, como las temporadas de clases en

colegios y escuelas, vacaciones y festividades, que son factores típicos a lo largo del año; con estos datos se realiza un aprendizaje de máquina, la cual será capaz de presentar los resultados de predicción requeridos por el usuario del sistema.

La base de datos de llamadas de emergencia que recibe el Servicio Integrado de Seguridad ECU 911, que se utiliza para el propósito mencionado, fue proporcionada por esa institución y en la cual se excluyen datos personales del autor de la llamada y de cualquier otra persona implicada en el incidente, por ser información confidencial. Además de que existe un convenio entre el Servicio Integrado de Seguridad ECU 911 y la Universidad de las Fuerzas Armadas ESPE para el intercambio de información con fines investigativos, gestionado por el Ing. Danilo Corral, colaborador de este trabajo de investigación (Anexo 1).

Así, teniendo esos datos aproximados y casi exactos, será posible para los servidores de atención de emergencias estar listos para realizar sus funciones inmediatamente y asegurar la integridad de los implicados en dicha emergencia.

1.3. Alcance del Proyecto

Realizar el preprocesamiento de los datos obtenidos por alertas de emergencia en el Servicio Integrado de Seguridad ECU911 en el año 2014 para obtener el número y el tipo de alertas de incidentes por cada día del año, para luego utilizarlos como datos de entrada en el aprendizaje de una máquina de vectores soporte (SVM), la cual utiliza técnicas de regresión para lograr dicho aprendizaje con un resultado con factor de predicción.

Para el número de incidentes se pretende realizar un conteo de las alertas registradas en la base de datos mencionada de acuerdo al mes, distrito, día de la semana, número de día de la semana dentro del mes y turno (horario) en que ocurrió cada incidente, para cada día de cada mes del 2014. En cuanto a los tipos de incidentes, se realizará un reconteo de acuerdo a las siete áreas de manejo de emergencias que registra el ECU 911 (gestión de siniestros, gestión de riesgos, gestión militar, seguridad ciudadana, servicio sanitario, servicios municipales y tránsito y movilidad) también para cada día del año en cuestión y en cada turno, y si se da el caso de que existen varios tipos de incidentes en un mismo turno, se tomará como objetivo al más probable, es decir, al que tiene más número de alertas.

También es fundamental realizar un análisis de las características que presentan los datos de entrada y así asegurar que se encuentren en las condiciones necesarias para alcanzar los mejores resultados en la regresión, pudiendo ser modificados con el objetivo de evaluar el aprendizaje de máquinas basado en parámetros de exactitud y precisión.

El aprendizaje de máquina se realiza mediante el software Matlab R2016a que posee las librerías necesarias para este fin y que además es útil para el preprocesamiento de los datos incluyendo técnicas necesarias de adecuación de los mismos para el correcto desempeño del aprendizaje de la máquina y su funcionamiento al final.

Para el uso de la máquina diseñada es necesaria una interfaz gráfica que cualquier usuario pueda manejar y entender, para el ingreso de datos y la visualización de resultados, la cual se realiza en el mismo Matlab. Además, se tiene que realizar un post procesamiento para que el resultado que devuelve la máquina sea entendido por cualquier persona que maneje la interfaz gráfica del software creado.

1.4. Objetivos

1.4.1. Objetivo General

Implementar un software para la predicción de incidentes que se alertan en un Servicio Integrado de Seguridad por cada centro de control que posea el país objetivo, diariamente, mediante el aprendizaje de máquinas.

1.4.2. Objetivos Específicos

- Investigar las bases y funcionamiento del aprendizaje de máquina, con énfasis en las máquinas de vectores soporte (SVM).
- Crear una base de datos con número y tipo de incidentes por día del año, en este caso a partir de los datos proporcionados por el Servicio Integrado de Seguridad ECU 911 del año 2014 para el centro de control Quito y procesar mediante Matlab esta nueva base de datos con técnicas de adecuación para su ingreso como datos de entrada y salida en el aprendizaje de máquina.
- Implementar dos modelos de máquinas de vectores soporte (SVM) con el número de incidentes y el tipo respectivamente para luego procesar en Matlab los datos de salida para la fácil comprensión del usuario que utilice el software.

- Estructurar una interfaz gráfica en Matlab para el registro de los respectivos datos de entrada en las máquinas creadas que son: distrito, mes, día y turno, y la visualización de los resultados respectivos.
- Evaluar el software creado, principalmente el desempeño de las máquinas mediante la técnica de evaluación *k-fold*.

CAPÍTULO II

2. FUNDAMENTO TEÓRICO

2.1. Aprendizaje de máquinas.

El aprendizaje de máquina es la disciplina que permite diseñar y desarrollar algoritmos para que un sistema sea capaz de generalizar comportamientos y reconocer patrones a partir de datos empíricos, experiencia y entrenamiento, y así también evolucionar y adaptarse a cambios que ocurren en su entorno.

Básicamente, se puede decir que la tarea del aprendizaje de máquinas es extraer conocimiento desde algunas propiedades o características no observadas de un determinado objeto basándose en las propiedades que si han sido observadas del mismo; o predecir el comportamiento futuro a partir de lo ocurrido en el pasado (Sancho Caparrini, 2017).

2.1.1. Tipos de aprendizaje.

Se pueden aplicar varios tipos de algoritmos de aprendizaje de acuerdo a la salida que se produzca en los mismos, y son:

- Aprendizaje supervisado:

En el cual se proveen ejemplos de aprendizaje que son pares de datos de entrada/salida de la forma (x_n, y_n) donde x_n son las posibles entradas (características) y y_n son las salidas (etiquetas). El objetivo es generar una función f que establezca una correspondencia entre las entradas y salidas, la cual es llamada clasificador si la salida es discreta y función de regresión si la salida es continua.

- Aprendizaje no supervisado:

  Los ejemplos de aprendizaje son datos de entrada solamente, sin las etiquetas correspondientes, por lo que el objetivo de este algoritmo es que el sistema sea capaz de reconocer patrones para determinar la salida o etiqueta correcta de los nuevos datos.

  Para lograr esto se asume que existe una estructura en el espacio de entrada, tal que ciertos patrones ocurren más seguido que otros y así saber qué sucede generalmente y que no.

- Aprendizaje semi-supervisado:

  Estos algoritmos usan datos etiquetados y no etiquetados, aunque los primeros suelen ser en menor cantidad. Se utilizan los datos no etiquetados para mejorar las tareas de aprendizaje supervisado cuando los datos etiquetados son escasos o costosos (Carrera Erazo, 2016).

De los cuales, en este caso se utiliza el aprendizaje supervisado ya que se cuenta con las etiquetas o respuestas para cada uno de los registros.

2.1.2. SVM.

Las SVM son un tipo de máquinas de aprendizaje con algoritmos de aprendizaje supervisado, que a diferencia de las demás, ésta no busca minimizar el error cometido por el modelo generado a partir de los datos de entrenamiento, sino minimizar el riesgo estructural, es decir compensar el error y el sobreajuste del modelo para lograr una mayor capacidad de generalización. Las SVM se utilizan para resolver problemas de clasificación binaria y regresión, por lo cual es necesario caracterizar cada uno por separado (Carmona Suárez, 2014).

2.1.2.1. SVM para clasificación binaria

Dentro de la tarea de clasificación pertenecen a los clasificadores lineales puesto que se utilizan separadores o hiperplanos lineales para diferenciar entre clases, y a partir de esto incrementa la complejidad y eficacia de estas máquinas, pudiendo haber ejemplos con clases: linealmente separables, linealmente cuasi-separables o no linealmente separables.

Pueden existir infinitos hiperplanos de separación, pero el objetivo de la SVM es hallar aquel (hiperplano óptimo) que equidista de los vectores de características más cercanos de cada clase y que maximiza el margen hacia ellos, estos se denominan vectores soporte; solo así se consigue una mejor generalización por parte del modelo.

La clasificación binaria no siempre es factible de forma simple mediante un hiperplano, así existe tres tipos de clasificación según la ubicación de los elementos pertenecientes

a cada clase y el método a utilizar para lograr la separación lineal. El primero corresponde a la clasificación de los elementos o registros de un conjunto de datos, linealmente separables, en el cual dado un conjunto separable de ejemplos $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$, donde $x_i \in R^d$ e $y_i \in \{+1, -1\}$, se puede definir un hiperplano de separación como una función lineal que es capaz de separar dicho conjunto sin error:

$$D(x) = (\omega_1 x_1 + \cdots + \omega_d x_d) + b = < \omega, x > + b$$

donde el vector ω (pesos) representa el vector normal del hiperplano, que indica su dirección, mientras que el coeficiente b indica su distancia al origen; por lo tanto, la SVM debe encontrar los valores de ω y b para hallar el hiperplano (Gálvez, 2014). El hiperplano debe cumplir:

$< \omega, x_i > + b \geq 0 \; si \; y_i = +1$

$< \omega, x_i > + b \leq 0 \; si \; y_i = -1, \; i = 1, \dots, n \quad o \quad y_i D(x_i) \geq 0, i = 1, \dots, n$ Sabiendo que la distancia entre el hiperplano y un ejemplo x' es: $|D(x')| \; / \; ||\omega||$, entonces:

$$y_i D(x_i) \geq \tau \; ||\omega||, \qquad i = 1, \dots, n$$

donde τ es el margen y su producto con $||\omega||$ se fija arbitrariamente a la unidad para llegar a la conclusión de que aumentar el margen equivale a disminuir la norma de ω: $\tau = 1/||\omega||$; de lo que se puede decir que para los ejemplos para los que $y_i D(x_i) = \tau \; ||\omega|| = 1$, son los vectores soporte; como se ve en la figura 1.

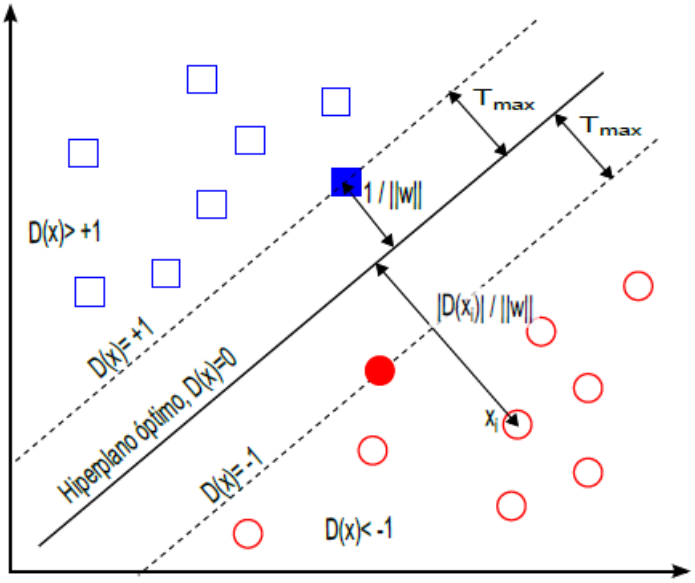

Figura 1 Ejemplos separables linealmente
(Carmona Suárez, 2014)

Así la búsqueda del hiperplano óptimo se formaliza como la obtención del mínimo de $||\omega||$:

$$mín\ f(\omega) = 1/2||\omega||^2 = 1/2 < \omega; \omega >$$

Este problema de optimización con restricciones corresponde a un problema de programación cuadrático y es abordable mediante la teoría de la optimización. Dicha teoría establece que un problema de optimización, denominado primal, tiene una forma dual si la función a optimizar y las restricciones son funciones estrictamente convexas. En estas circunstancias, resolver el problema dual permite obtener la solución del problema primal.

Finalmente se debe utilizar la resolución del problema dual del Lagrangiano sujeto a las condiciones de que a cada lado del hiperplano se encuentren miembros de la misma

clase, considerando cuánto interesa realizar correctamente la clasificación de los datos, para encontrar los valores de ω y b:

$$\omega^* = \sum_{i=1}^{n} \alpha_i y_i x_i \,, \qquad i = 1, \dots, n$$

$$b^* = x_{vs} - < \omega^*; x_{vs} >$$

Se da que los multiplicadores de Lagrange son nulos para todo vector de características que no pertenezca al conjunto de vectores de soporte.

El segundo tipo de clasificación binaria corresponde a la clasificación de registros linealmente cuasi-separables, que corresponde a los elementos ruidosos o dispersos aleatoriamente que son los que se encuentran comúnmente en la vida real, por lo que ahora la estrategia es relajar el grado de separabilidad del conjunto de ejemplos permitiendo que haya errores de clasificación en algunos de ellos, pero sigue siendo el objetivo encontrar el hiperplano óptimo para el resto de ejemplos que si son separables.

En este caso no se cumple que $y_i D(x_i) \geq 1$ por lo que se pueden dar dos casos, el primero en que el ejemplo cae dentro del margen del lado de la clase correcta del hiperplano de separación y el segundo en que el ejemplo cae al otro lado, siendo ambos casos no separables pera el primero clasificado de forma correcta y el segundo no.

Para abordar este nuevo problema se introduce una variable de holgura ξ_i, $i = 1, \dots, n$, en la definición del hiperplano de separación, la cual representa la desviación del caso separable, medida desde el borde del margen que corresponde a la clase y_i, como se ve en la figura 2:

$$y_i D(x_i) \geq 1 - \xi_i$$

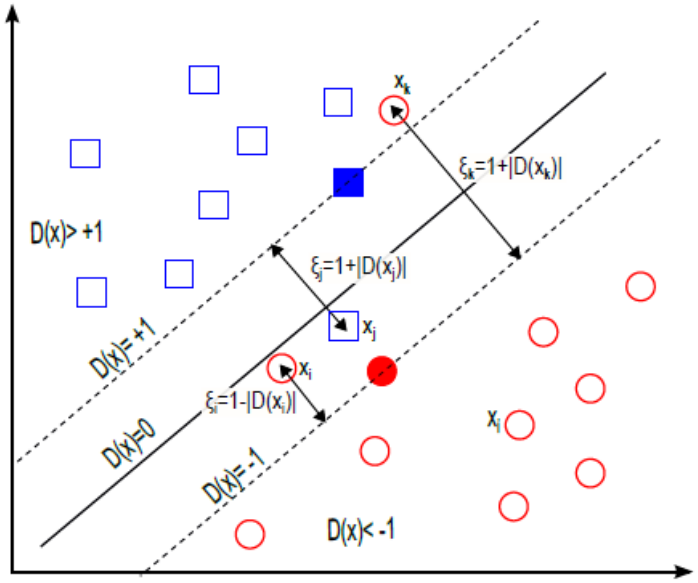

Figura 2 Ejemplo cuasi-separable linealmente
(Carmona Suárez, 2014)

Entonces, variables de holgura de valor cero corresponden a ejemplos separables, mayores que cero corresponden a ejemplos no separables y mayores que uno corresponden a ejemplos no separables y, además, mal clasificados. Por tanto, la suma de todas las variables de holgura, $\sum_{i=1}^{n} \xi_i$, permite, de alguna manera, medir el coste asociado al número de ejemplos no-separables.

Ahora es necesario incluir los errores de clasificación que está cometiendo el hiperplano de separación, dentro de la optimización:

$$mín \;\; f(\omega) = 1/2 < \omega; \omega > + C \sum_{i=1}^{n} \xi_i$$

donde C es una constante, suficientemente grande, elegida por el usuario, que permite controlar en qué grado influye el término del coste de ejemplos no-separables en la minimización de la norma, es decir, permitirá regular el compromiso entre el grado de

sobreajuste del clasificador final y la proporción del número de ejemplos no separables. Así, un valor de C muy grande permitiría valores de ξ_i muy pequeños. En el límite ($\mathrm{C} \to \infty$), estaríamos considerando el caso de ejemplos perfectamente separables ($\xi_i \to 0$). Por contra, un valor de C muy pequeño permitiría valores de ξ_i muy grandes, es decir, estaríamos admitiendo un número muy elevado de ejemplos mal clasificados. En el caso límite ($\mathrm{C} \to 0$), se permitiría que todos los ejemplos estuvieran mal clasificados ($\xi_i \to \infty$).

Igual al caso separable linealmente, el problema de optimización se resuelve en su forma dual, resultando:

$$\omega^* = \sum_{i=1}^{n} \alpha_i y_i x_i$$

Antes de obtener una expresión para el cálculo del valor de b^*, se considerarán algunos resultados interesantes, se puede afirmar que todos los ejemplos x_i cuyo α_i asociado sea igual a cero corresponden a ejemplos separables ($\xi_i = 0$), que todos los ejemplos x_i cuyo $\alpha_i = C$ corresponderán a ejemplos no-separables ($\xi_i > 0$).

Se puede entonces considerar los dos casos que se muestran en la figura 2, el primero, el ejemplo, x_i, aunque no separable, está bien clasificado, es decir, $y_i D(x_i) \geq 0$, entonces $\xi_i = 1 - |D(x_i)|$. En el segundo caso, el ejemplo, x_i, es no separable y está mal clasificado, es decir, $y_i D(x_i) < 0$, entonces $\xi_i = 1 + |D(x_i)|$..

Finalmente, se puede afirmar que un ejemplo, x_i, es vector soporte si y solo si $0 < \alpha_i < C$, y se tiene que el valor de b^* es:

$$b^* = y_i \sum_{i=1}^{n} \alpha_i{}^* y_i < x_j, x_i > \forall\, \alpha_i \;\; t.q. \, 0 < \alpha_i < C$$

Finalmente, el tercer tipo de clasificación binaria es aquel que tiene los registros linealmente no separables; en los casos anteriores el hiperplano óptimo se definía como función lineal en el espacio de ejemplos, pero cuando los ejemplos son no separables linealmente se deben usar funciones base, no lineales, para definir espacios transformados de alta dimensionalidad (espacio de características) y allí buscar el hiperplano lineal óptimo de separación. Este hiperplano será no lineal en el espacio original de los ejemplos.

Sea $\Phi: X \to \mathbb{F}$ la función de transformación que hace corresponder cada vector de entrada x con un punto en el espacio de características $\mathbb{F}$, donde $\Phi(x) = [\Phi_1(x), \dots, \Phi_m(x)]$ y $\exists \Phi_i(x);\ i = 1, \dots, m$, tal que $\Phi_i(x)$ es una función no lineal. En este contexto, la función de decisión en el espacio de características vendrá dada por:

$$D(x) = (\omega_1 - \Phi_1(x) + \cdots + \omega_m - \Phi_m(x)) =< \omega, \Phi(x) >$$

y, en su forma dual, la función de decisión se obtiene transformando convenientemente la expresión de la frontera de decisión en:

$$D(x) = \sum_{i=1}^{n} \alpha_i{}^* y_i K(x, x_i)$$

donde $K(x; x')$ se denomina función kernel.

Una función kernel es una función $K: X \times X \to R$ que asigna a cada par de elementos del espacio de entrada, X, un valor real correspondiente al producto escalar de las imágenes de dichos elementos en un nuevo espacio $\mathbb{F}$ (espacio de características), es decir:

$$K(x; x') =< \Phi(x), \Phi(x') >= (\Phi_1(x)\ \Phi_1(x')) + \cdots + \Phi_m(x)\ \Phi_m(x'))$$

Así el nuevo problema dual a resolver es:

$$máx \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{s. a.} \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n$$

Las funciones kernel más comunes son:

- Kernel lineal:

$$K(x, x') = < x, x' >$$

- Kernel polinómico de grado *p*:

$$K_p(x, x') = [\sigma < x, x' > + \tau]^p$$

- Kernel gaussiano:

$$K(x, x') = exp(-\sigma ||x, x'||^2), \sigma > 0$$

- Kernel sigmoidal:

$$K(x, x') = tanh(\sigma < x, x' > + \tau)$$

2.1.2.2. SVM para regresión

Dado el mismo conjunto de ejemplos de entrenamiento $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, donde $x_i \in R^d$ e $y_i \in R$, en el caso de la regresión se asume que los valores y_i de todos los ejemplos de S se pueden ajustar o cuasi-ajustar mediante una función lineal, para lo cual se deben encontrar los pesos $(\omega_1 x_1 + \cdots + \omega_d x_d)$ y poder definir la misma:

$$f(x) = (\omega_1 x_1 + \cdots + \omega_d x_d) + \ b =< \omega, x > + b$$

Es necesario permitir cierto ruido para los ejemplos de entrenamiento relajando la condición de error entre el valor predicho por la función lineal y el valor real, mediante la función de pérdida ϵ:

$$L_\epsilon\big(y, f(x)\big) = \begin{cases} 0 & ; si\ |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon & ; \quad en\ otro\ caso \end{cases}$$

Entonces, contrario al caso de clasificación, los vectores soporte son aquellos que definen la frontera externa de una región tubular limitada por $\pm\epsilon$ y todo ejemplo confinado en el interior se considera parte de la función lineal, como se ve en la figura 3.

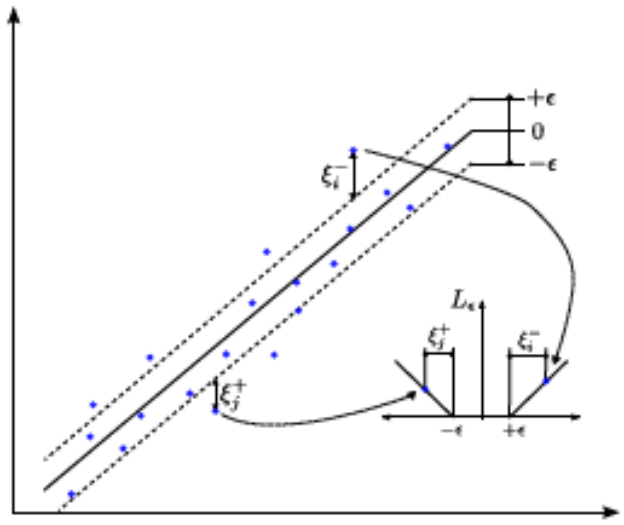

Figura 3 SVM para regresión
(Carmona Suárez, 2014)

Debido que en los casos reales no todos los ejemplos se ajustan al modelo lineal, se deben agregar las variables de holgura $\xi_i{}^+$ y $\xi_i{}^-$, para cuantificar la magnitud de este error de predicción. Así, la variable $\xi_i{}^+ > 0$ cuando la predicción del ejemplo es mayor

que su valor real, es decir, $f(x_i) - y_i > \epsilon$, será $\xi_i^- > 0$ cuando $y_i - f(x_i) > \epsilon$ y será igual a cero en otros casos.

La función a optimizar es similar al caso de clasificación, pero con las dos variables de holgura:

$$\text{mín } 1/2 < \omega; \omega > + \mathrm{C} \sum_{i=1}^{n} (\xi_i^+ + \xi_i^-)$$

$$s.a. \quad (< \omega, x_i > + b) - y_i - \epsilon - \xi_i^+ \leq 0$$

$$y_i - (< \omega, x_i > + b) - \epsilon - \xi_i^- \leq 0$$

$$\xi_i^+, \xi_i^- \geq 0, i = 1, \ldots, n$$

Finalmente, la transformación al problema dual se obtiene mediante la función Lagrangiana, y de esta se tienen los valores:

$$\omega = \sum_{i=1}^{n} (\alpha_i^- - \alpha_i^+) \mathrm{x}_i$$

$$b^* = \mathrm{y}_i - < \omega^*, \mathrm{x}_i > + \epsilon \; si \; 0 \leq \alpha_i^+ \leq C$$

$$b^* = \mathrm{y}_i - < \omega^*, \mathrm{x}_i > - \epsilon \; si \; 0 \leq \alpha_i^- \leq C$$

Ahora, cuando la función lineal no ajusta los ejemplos de entrada, se procede de forma similar al caso de clasificación no separable linealmente, transformando estos ejemplos del espacio original a un espacio de características de mayor dimensión, en el que si se ajustan linealmente. La función lineal se estructura de acuerdo al kernel utilizado con el siguiente modelo:

$$f(x) = \sum_{i=1}^{n} (\alpha_i^- - \alpha_i^+) K(\mathrm{x}, \mathrm{x}_i)$$

Los coeficientes α_i^-, α_i^+ se obtienen como resultado de resolver el problema dual, expresado ahora como:

$$\text{máx} \sum_{i=1}^{n} (\alpha_i^- - \alpha_i^+) y_i - \epsilon \sum_{i=1}^{n} (\alpha_i^- + \alpha_i^+) - \frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i^- - \alpha_i^+)(\alpha_j^- - \alpha_j^+) K(x_i, x_j)$$

$$\text{s.a.} \sum_{i=1}^{n} (\alpha_i^+ - \alpha_i^-) = 0$$

$$0 \leq \alpha_i^+, \alpha_i^- \leq C, \qquad i = 1, \dots, n$$

Se debe elegir el kernel, ϵ y C más adecuado, considerando que, en problemas de regresión con ruido, el parámetro ϵ debería ser elegido de forma que refleje la varianza del ruido de los datos, problemas de regresión sin ruido (problemas de interpolación) el valor ϵ corresponde a la exactitud preestablecida de interpolación, de forma que, cuanto mayor sea el valor de ϵ, menor número de vectores soporte se necesitarán, y viceversa. Por otro lado, la metodología usada para seleccionar el valor de C más adecuado, se basa normalmente en técnicas de validación cruzada (Carmona Suárez, 2014).

2.1.3. Evaluación de máquinas de aprendizaje.

Los clasificadores son entrenados con un conjunto de entrenamiento finito y deben ser probados con un conjunto de prueba diferente experimentalmente, así el rendimiento del clasificador con los datos de prueba es un aproximado del rendimiento con datos nunca vistos.

La idea principal es revisar la habilidad de generalización del clasificador, ya que si es demasiado preciso usualmente presenta resultados pobres con datos nuevos. Mientras

el clasificador tiene más datos de entrenamiento, presenta mejor generalización y mientras más datos de prueba, mejor estimación de la probabilidad de error de clasificación.

Los criterios de evaluación del rendimiento de un clasificador son: tasa de error o exactitud, la matriz de confusión y el riesgo bayesiano esperado.

La técnica usada para dividir el conjunto de datos finito es:

- Validación cruzada *k-fold*: El conjunto de datos total se divide aleatoriamente en *k* subconjuntos disjuntos de igual tamaño donde cada uno tiene aproximadamente la misma distribución de clase. El clasificador es entrenado *k* veces, tomando cada vez un subconjunto diferente como conjunto de prueba; el error estimado es la madia de los *k* errores. Se recomienda usar $k = 5$ o $k = 10$.

2.2. Matlab y SVM.

La implementación del aprendizaje de máquinas para hallar patrones o predecir futuros resultados en el comportamiento de un sistema que maneja grandes cantidades de datos, se debe realizar mediante un software capaz de ello, para lo que Matlab posee librerías especializadas a partir de la versión R2014a y ha ido evolucionando hasta el actual R2016b incorporando funciones más eficientes, dentro del *Statistics and Machine Learning Toolbox*, para clasificación y regresión que permiten describir, analizar y modelar datos.

Este *toolbox* también permite realizar selección de las características relevantes para el modelado, mediante diferentes métodos de reducción como PCA (análisis de componentes principales).

El Statistics and Machine Learning Toolbox incluye algoritmos de aprendizaje de máquinas supervisados o no supervisados como son: máquinas de vectores soporte (SVM), árboles de decisión, k vecino más cercano, Bayes, redes neuronales, etc.

En este caso se utilizan las herramientas de SVM para clasificación y regresión en la versión de Matlab R2016a que incluye las funciones: *fitrsvm, fitcsvm, fitrlinear, fitclinear, fitcecoc* y *predict* que son las que se utilizan, y son versiones más eficientes de otras que presentaban versiones anteriores de Matlab.

La función *fitcecoc*, que provee un modelo de clasificador multiclase con códigos de salida de corrección de errores (ECOC) de un conjunto de datos predictores específico, usando modelos SVM binarios $K(K-1)/2$ donde K es el número de etiquetas de clase únicas, con diseño de aprendizaje uno contra uno, es decir, que se realiza el aprendizaje binario entre pares de clases, todas contra todas. Sin embargo, esta función no se utilizará en el desarrollo del modelo SVM ya que se busca predecir un resultado aproximado y no exacto mediante regresión con los datos de números decimales, pero sí clasificación con los datos de números binarios (Matlab R2016a, 2016).

2.2.1. Función *fitrsvm* y *fitcsvm*

Son funciones que proveen un modelo de regresión (*fitrsvm*) o un modelo de clasificación binario (*fitcsvm*) de máquina de vectores soporte para un conjunto de datos

predictor de baja o moderada dimensión. Soportan el mapeo de datos predictores usando funciones kernel y SMO (Optimización Mínima Secuencial), ISDA (Algoritmo Iterativo de Datos Único) o minimización suave de márgenes L1 para resolver el problema de QP (programación cuadrática) y a través de esta realizar la minimización de la función objetivo.

La sintaxis para el uso de estas funciones es:

Mdl = fitrsvm(X,Y,Name,Value) 'o' Mdl = fitcsvm(X,Y,Name,Value)

Donde *X* es la matriz de valores predictores (registros con sus características) y *Y* es el vector con los valores de respuesta en el caso de regresión y las etiquetas de las clases en el caso de clasificación; estos valores deben ser del tipo *single* o *double. Name-Value* es un par de argumentos para opciones adicionales (si se requiere) como por ejemplo una función kernel específica o el tipo de evaluación para la SVM.

Las matrices *X* y *Y* pueden ser reemplazadas por una tabla, especificando el nombre la columna de respuesta en lugar de *Y* respectivamente (Matlab R2016a, 2016).

2.2.2. Función *fitrlinear* y *fitclinear*

Estas funciones entrenan modelos SVM de regresión (*fitrlinear*) o de clasificación binaria (*fitclinear*) de manera eficiente con datos predictores dispersos o completos de gran dimensión. Minimiza la función objetivo usando técnicas que reducen el tiempo de cálculo, por lo que disminuye el uso de recursos computacionales y el proceso es relativamente rápido.

La sintaxis para el uso de estas funciones es:

Mdl = fitrlinear(X,Y,Name,Value) 'o' Mdl = fitclinear(X,Y,Name,Value)

donde *X* es la matriz de valores predictores (registros con sus características) y *Y* es el vector con los valores de respuesta en el caso de regresión y las etiquetas de las clases en el caso de clasificación; estos valores deben ser del tipo *single* o *double*. *Name-Value* es un par de argumentos para opciones adicionales (si se requiere) como por ejemplo especificación para cross-validate o el tipo de regularización para la SVM.

Las matrices *X* y *Y* pueden ser reemplazadas por una tabla, especificando el nombre la columna de respuesta en lugar de *Y* respectivamente (Matlab R2016a, 2016).

2.2.3. Función predict

Para predecir la respuesta de un registro basándose en el modelo de una máquina de aprendizaje, en este caso a través de las funciones descritas anteriormente.

La sintaxis para el uso de estas funciones es:

pred = predict(Mdl,X)

donde *Mdl* es el modelo de la máquina de aprendizaje y *X* el vector de datos que se quiere predecir (Matlab R2016a, 2016).

CAPÍTULO III

3. IMPLEMENTACIÓN DEL MODELO DE PREDICCIÓN DE INCIDENTES

3.1. Extracción de datos para la creación de nuevos registros.

Las bases de datos proporcionadas por el Servicio Integrado de Seguridad ECU 911 son del año 2014 del centro de operaciones de la ciudad de Quito, están divididas por cada mes del año, es decir un archivo Excel de cada uno, los cuales contienen en promedio 85000 registros de incidentes con diferentes características cada uno, las cuales se enlistan en la Tabla 1.

Dentro del territorio ecuatoriano se encuentran operativos 15 centros zonales del ECU 911, el de Quito específicamente cubre las provincias de Pichincha, Orellana y Napo (El Ciudadano, 2014), y cada una se divide en distritos, teniendo en total 17 que son: Aguarico, Calderón, Cayambe, el valle de Quijos, Eloy Alfaro, Eugenio Espejo, La Delicia, La Joya de los Sachas, Los Chillos, Manuela Sáenz, Nanegal, Noroccidente, Orellana Loreto, Quitumbe, Rumiñahui-Mejía, Tena y Tumbaco; entonces se necesita saber el número y el tipo de incidentes reportados en llamadas al ECU 911 a partir de cada base de datos mensual.

Tabla 1

Características de la base de datos original

Características	Descripción
Centro	Centro zonal al que pertenece
Código	Individual para cada incidente
Año del incidente	Año de ocurrencia
Mes del incidente	Mes de ocurrencia
Semana del incidente	Semana de ocurrencia
Día del incidente	Día de ocurrencia
Fecha del incidente	Fecha completa de ocurrencia
Hora del incidente	Hora de ocurrencia
Tipo de solicitud del incidente	Por qué medio se dio a conocer la emergencia
Invalidación del incidente	Tipo de invalidación del incidente
Dirección del incidente	Dirección de ocurrencia
Coordenadas de longitud y latitud	Coordenadas del lugar de ocurrencia
Región	Región en que se reporta el incidente
Zona	Zona en que se reporta el incidente
Provincia	Provincia en que se reporta el incidente
Cantón	Cantón en que se reporta el incidente
Distrito	Distrito en que se reporta el incidente
Circuito	Circuito en que se reporta el incidente
Subcircuito	Subcircuito en que se reporta el incidente
Puesto del evaluador	Extensión telefónica del puesto del evaluador que atendió el incidente
Turno	Turno en que se atendió el incidente
Descripción del incidente	Dirección y descripción de la situación reportada
Tipo de incidente	Tipo de incidente de acuerdo a la institución encargada de atenderlo
Grado del incidente	Nivel de emergencia de acuerdo a un código de colores (verde, amarillo, naranja o rojo)
Hora de inicio	Hora en que inicio el incidente
Verificador	1 si fue como se reportó y 0 si no fue así
Hora de atención del incidente	Hora en que se atendió el incidente
Institución	Institución que atendió el incidente de acuerdo al tipo del mismo
Descripción de la tarea realizada	Descripción de cómo se atendió el incidente
Tipo de respuesta	Tipo de vehículo para atender la emergencia
Acción de fin del incidente	Como finalizó la atención del incidente
Hora de fin	Hora en que finalizó la atención del incidente

De todas las características que poseen las bases de datos se puede decir que solo son relevantes, para la predicción del número y tipo de incidentes, el distrito, mes, día del incidente, fecha del incidente, turno (horario) que puede ser de 00:00 a 06:00, 06:00 a 12:00, 12:00 a 18:00 o 18:00 a 00:00 y tipo de incidente, ya que las demás no aportan en el conteo del número y tipo incidentes a simple vista, además de que tomar en cuenta circuito o subcircuito significa desglosar más los registros y por lo tanto conseguir menos eficiencia a la hora de procesar datos y menor capacidad de generalización del modelo SVM; entonces es necesario armar una nueva base de datos con dichas características de entrada y el número de incidentes que se produjeron en cada uno y otra con las mismas características de entrada pero el tipo de incidente más probable en ese turno como etiqueta; todo esto para los 365 días del año en cada distrito.

Para armar la base de datos del tipo de incidentes se debe saber todos los tipos que se produjeron en cada turno del día y cuantas veces cada uno, así el que se haya suscitado más veces será el más probable y el que se escoge como salida de cada registro para dicha base de datos nueva.

3.1.1. Algoritmo para la extracción de datos.

Para iniciar la construcción de las nuevas bases de datos, en formato de tablas, se enlista cada registro o ejemplo en una fila y sus respectivas características por columnas iniciando con el mes, el distrito, el día de la semana y turno, y para cada registro su etiqueta de salida que para la primera base de datos nueva es el conteo de los incidentes que se produjeron y para la segunda el tipo de incidente más probable.

Entonces se crea un algoritmo capaz de contar el número de incidentes para las características mencionadas a partir de las bases de datos mensuales originales y otro para contar cuantos incidentes se produjeron de cada uno de los siete posibles tipos que son: gestión de siniestros, gestión de riesgos, gestión militar, seguridad ciudadana, servicio sanitario, servicios municipales y tránsito y movilidad, para finalmente implementarlos en Matlab.

3.1.2. Estructuración de registros a partir de los datos extraídos.

En el caso del número de incidentes, cada vez que se ejecuta el algoritmo creado en Matlab es necesario primero importar cada base de datos mensual, que está en formato xls (Excel), en Matlab como una tabla y especificar el distrito para el cual se quiere realizar el conteo dentro del código (Anexo 2), el cual busca de uno en uno cada día del mes y el turno del día, cuenta cuantos registros coinciden en dicho día y turno y guarda ese resultado en un vector numérico que tiene el siguiente formato: todos los elementos corresponden al mismo mes y mismo distrito pero cada cuatro elementos corresponden a un mismo día de la semana y de ellos cada uno a un turno diferente de ese día.

Entonces la nueva base de datos nueva completa tendrá ese formato, quedando como se ve en el ejemplo de los primeros tres días de diciembre del distrito Calderón en la Tabla 2.

Tabla 2

Ejemplo del formato de la nueva base de datos de número de incidentes

Mes	Distrito	Dia_Semana	Turno	Num_Incidentes
Diciembre	CALDERON	Lunes	00-06	19
Diciembre	CALDERON	Lunes	06-12	55
Diciembre	CALDERON	Lunes	12-18	51
Diciembre	CALDERON	Lunes	18-00	52
Diciembre	CALDERON	Martes	00-06	5
Diciembre	CALDERON	Martes	06-12	33
Diciembre	CALDERON	Martes	12-18	28
Diciembre	CALDERON	Martes	18-00	62
Diciembre	CALDERON	Miercoles	00-06	19
Diciembre	CALDERON	Miercoles	06-12	20
Diciembre	CALDERON	Miercoles	12-18	28
Diciembre	CALDERON	Miercoles	18-00	46

En el caso del tipo de incidentes, cada vez que se ejecuta el algoritmo creado en Matlab también es necesario importar cada base de datos mensual, que está en formato xls (Excel), en Matlab como una tabla y especificar dentro del código (Anexo 3) el distrito para el cual se quiere realizar el conteo de cada tipo de incidentes y obtener el mayor de ellos, lo cual se logra buscando de uno en uno cada día del mes, turno del día y tipos de incidentes, cuenta los registros que coinciden con cada uno de los siete posibles tipos y guarda en un vector numérico el mayor de ellos, donde cada elemento es el código de un tipo específico, como se ve en la Tabla 3, cuando todos son cero el valor guardado es cero y cuando dos valores coinciden escoge el primero de ellos, ya que solo el 4.9% presentan este caso y por lo tanto se tendría una influencia mínima en el rendimiento del modelo SVM final; el algoritmo devuelve un vector numérico en el mismo formato del vector del número de incidentes.

Entonces la segunda nueva base de datos nueva completa tendrá ese formato, quedando como se ve en el ejemplo de los primeros tres días de diciembre del distrito Calderón en la Tabla 4.

Tabla 3

Código para cada tipo de incidente

Tipo de Incidente	Código
Gestión de siniestros	1
Gestión de riesgos	2
Gestión militar	3
Seguridad ciudadana	4
Servicio sanitario	5
Servicios municipales	6
Tránsito y movilidad	7
No hay incidentes	0

Tabla 4

Ejemplo del formato de la nueva base de datos de tipo de incidentes

Mes	Distrito	Dia_Semana	Turno	Tipo_Incidentes
Diciembre	CALDERON	Lunes	00-06	5
Diciembre	CALDERON	Lunes	06-12	4
Diciembre	CALDERON	Lunes	12-18	4
Diciembre	CALDERON	Lunes	18-00	4
Diciembre	CALDERON	Martes	00-06	5
Diciembre	CALDERON	Martes	06-12	7
Diciembre	CALDERON	Martes	12-18	4
Diciembre	CALDERON	Martes	18-00	4
Diciembre	CALDERON	Miercoles	00-06	4
Diciembre	CALDERON	Miercoles	06-12	4
Diciembre	CALDERON	Miercoles	12-18	4
Diciembre	CALDERON	Miercoles	18-00	4

3.2. Preparación de los datos para el ingreso en el aprendizaje de máquina.

La aplicación de las funciones de SVM en Matlab requiere que los datos de entrada cumplan las condiciones mencionadas en la sección *Matlab y SVM*, por lo que se debe hacer un preprocesamiento de las bases de datos con los nuevos registros, principalmente convertir los datos en numéricos y después aplicar cualquier tipo de proceso o codificación.

3.2.1. Procesamiento de datos de entrada del modelo SVM.

3.2.1.1. Conversión a datos numéricos.

La conversión de todos los datos de las nuevas bases a números se realiza mediante la creación de un algoritmo para ejecutar en Matlab (Anexo 4), en el que se ingresa toda la tabla de la base de datos como parámetro de entrada, busca de registro en registro en la característica de mes del año de acuerdo a las tres primeras letras y les asigna el número, luego busca los registros en la característica de los distritos y les asigna un número, después busca los registros en las característica de día de la semana y les asigna un número, y finalmente busca en la característica del turno del día y le asigna el respectivo número, para devolver la misma entrada pero en valores de tipo *double.*

Se utiliza la siguiente codificación para la nueva tabla de la base de datos de salida, sea la del número de incidentes o la del tipo de incidentes: los meses en números del 1

al 12 respectivamente, los distritos del 1 al 17 respectivamente en orden alfabético, los días de la semana del 1 al 7 respectivamente y los turnos del día del 1 al 4 desde el turno que inicia a las 00:00 hasta el que termina a esa misma hora, quedando como se ve en el ejemplo de los primeros tres días de diciembre del distrito Calderón en la Tabla 5.

Tabla 5

Ejemplo de la conversión numérica de las nuevas bases de datos

Mes	Distrito	Dia_Semana	Turno	Num_Incidentes
12	2	1	1	19
12	2	1	2	55
12	2	1	3	51
12	2	1	4	52
12	2	2	1	5
12	2	2	2	33
12	2	2	3	28
12	2	2	4	62
12	2	3	1	19
12	2	3	2	20
12	2	3	3	28
12	2	3	4	46

3.2.1.2. Cuantificación de los valores de respuesta

En el caso de la base de datos con los nuevos registros del número de incidentes, los valores de respuesta o etiquetas (columna *Num_Incidentes*) son valores numéricos entre 0 y 1570, es decir se tienen 1571 posibilidades o clases, un rango muy amplio que puede causar un déficit de aprendizaje de la SVM principalmente al momento de aumentar la dimensión del espacio de características mediante la función kernel para encontrar el

hiperplano correcto de separación entre cada par de clases, provocando confusión entre estas.

La solución al problema mencionado es reducir el rango de los valores de respuesta aplicando cuantificación en un nivel razonable para no perder información y poder reescalar al nivel original después. Entonces se crea un algoritmo en Matlab (Anexo 5) que realiza una cuantificación uniforme de la siguiente manera: primero se divide el rango de valores de respuesta original en tantos niveles de cuantificación como se requiera (valor máximo original para número de niveles menos uno) y a cada valor se le asigna un número del cero al nivel máximo de cuantificación menos uno, luego se compara el valor original que se quiere cuantificar con sus dos valores, inmediato superior e inmediato inferior del rango mencionado, se asigna al más cercano, y por lo tanto al número de nivel de cuantificación que le corresponde; permite elegir el número de niveles como un parámetro de entrada, que por lo general es un valor potencia de dos para poder realizar cualquier codificación posteriormente. Este procedimiento se hace para cada una de las respuestas de la base de datos y al final se obtiene un vector con todas cuantificadas.

3.2.1.3. Escalado exponencial de los valores de respuesta

Por el hecho de que los valores de respuesta de la base del número de incidentes que se encuentran en la tercera parte más alta del rango total son muy pocos y del resto no existen tampoco valores en todos los niveles de cuantificación, hay muchos niveles que no se utilizan, por lo cual puede ser beneficioso en el aprendizaje reducir el número de

niveles dentro del mismo rango, pero dándole prioridad a los niveles más bajos, aplicando una escala exponencial a los datos ya cuantificados.

El resultado gráfico de los niveles de este escalado se puede observar en la Figura 4, con un ejemplo de 64 niveles de cuantificación uniforme en la gráfica azul y el mismo rango, pero escalado exponencialmente, en la gráfica roja. Este proceso se realiza con un algoritmo en Matlab (Anexo 6) el cual primero arma un vector con los valores de escala exponencial dependiendo del número de niveles de cuantificación uniforme hecha anteriormente y después asigna cada valor a los niveles de cuantificación uniforme más cercanos.

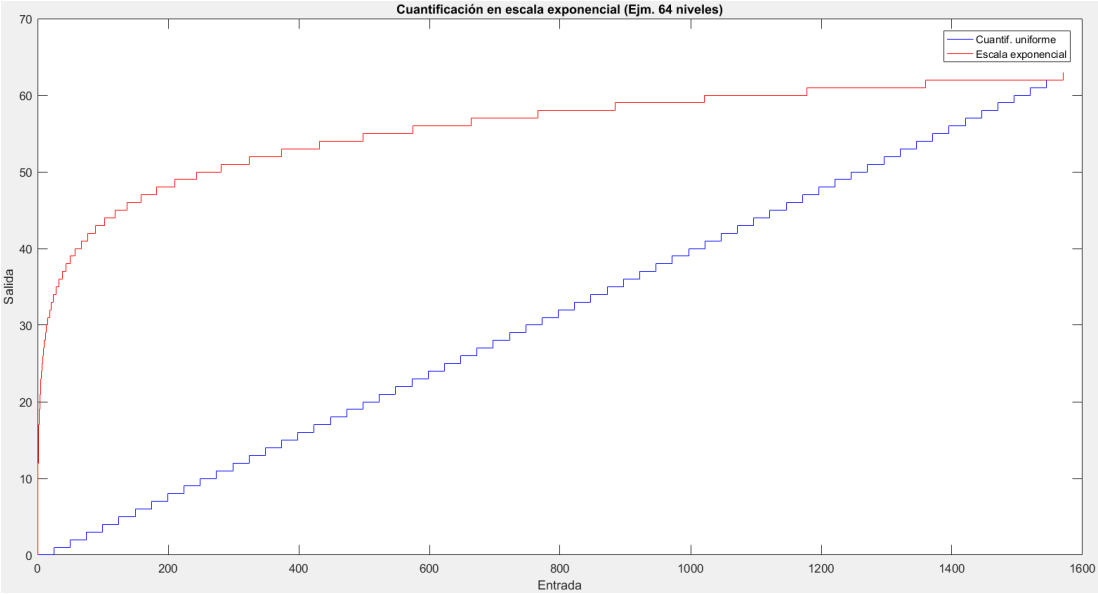

Figura 4 Ejemplo cuantificación y escalado a 64 niveles

3.2.1.4. Codificación binaria de los valores de respuesta

Una predicción exacta, con los mismos valores de respuesta de los registros de entrada, puede lograrse aplicando una clasificación binaria, que generalmente usa las clases 0 y 1. Para esto se puede realizar una codificación binaria de los valores de respuesta, ya sean los cuantificados uniformemente o los escalados exponencialmente.

Se transforma cada valor de respuesta en su respectivo número binario, todos con el mismo número de dígitos dependiendo del número de niveles de cuantificación, así por ejemplo para 64 niveles se utilizan 6 dígitos ($64 = 2^6$). Entonces se tienen tantos valores de respuesta como dígitos para cada registro de entrada y para cada columna de valores de entrada se debe modelar una SVM de clasificación, para el caso del ejemplo se tienen siete SVM's al final.

El código del algoritmo en Matlab que realiza la función de codificación binaria (Anexo 7) mediante la función *decimalToBinaryVector, que* se aplica a la tabla de la base de datos de número de incidentes ya cuantificada o escalada, es decir después del código respectivo o a la tabla de la base de datos del tipo de incidentes directamente.

3.2.2. Estructuración de una nueva base de datos del año 2014 procesada.

Después del procesamiento de los datos de las bases originales, se tienen cuatro posibles bases de datos nuevas listas para la experimentación de aprendizaje de máquinas mediante SVM con las funciones propias de Matlab y así poder evaluar cada modelo creado; estas bases pueden ser: base con datos numéricos (Tabla 5), base

numérica con valores de respuesta cuantificados uniformemente a 64 niveles como se ve en el ejemplo de la Tabla 6 de la primera semana de diciembre para el distrito Calderón, base numérica con respuesta cuantificada y escalada exponencialmente que se puede ver en el mismo ejemplo de la Tabla 7 y base numérica con respuesta cuantificada uniformemente y con codificación binaria como se ve en la Tabla 8 del mismo ejemplo.

Tabla 6

Ejemplo de la cuantificación uniforme de la nueva base de datos

Mes	Distrito	Dia_Semana	Turno	Num_Incidentes
12	2	1	1	1
12	2	1	2	2
12	2	1	3	2
12	2	1	4	2
12	2	2	1	0
12	2	2	2	1
12	2	2	3	1
12	2	2	4	2
12	2	3	1	1
12	2	3	2	1
12	2	3	3	1
12	2	3	4	2

Tabla 7

Ejemplo de la cuantificación y escalado exponencial de la nueva base de datos

Mes	Distrito	Dia_Semana	Turno	Num_Incidentes
12	2	1	1	32
12	2	1	2	40
12	2	1	3	39
12	2	1	4	39
12	2	2	1	23
12	2	2	2	36
12	2	2	3	35
12	2	2	4	40
12	2	3	1	32
12	2	3	2	33
12	2	3	3	35
12	2	3	4	38

Tabla 8

Ejemplo de la cuantificación y codificación de la nueva base de datos

Mes	Distrito	Dia_Semana	Turno	Num_Incidentes
12	2	1	1	0 0 0 0 0 1
12	2	1	2	0 0 0 0 1 0
12	2	1	3	0 0 0 0 1 0
12	2	1	4	0 0 0 0 1 0
12	2	2	1	0 0 0 0 0 0
12	2	2	2	0 0 0 0 0 1
12	2	2	3	0 0 0 0 0 1
12	2	2	4	0 0 0 0 1 0
12	2	3	1	0 0 0 0 0 1
12	2	3	2	0 0 0 0 0 1
12	2	3	3	0 0 0 0 0 1
12	2	3	4	0 0 0 0 0 1

3.3. Característica complementaria

Adicionalmente se requiere comprobar el nivel de información que puede aportarse al proceso de aprendizaje del modelo SVM, añadiendo una característica complementaria

a la del día de la semana, que es número de día de la semana dentro del mes, es decir, por ejemplo, el primer lunes del mes tiene asignado el número 1 y así como corresponde hasta máximo el número cinco.

La característica adicional quedaría como se ve en el ejemplo de la Tabla 9 y a la cual se realiza el mismo procesamiento descrito en las subsecciones anteriores para poder implementar el modelo SVM para el numero y el tipo de incidentes.

Tabla 9

Ejemplo del formato de la nueva base de datos de tipo de incidentes con la característica complementaria.

Mes	Distrito	Dia_Semana	Turno	Dia_Mes	Num_Incidentes
Diciembre	CALDERON	Lunes	00-06	1	19
Diciembre	CALDERON	Lunes	06-12	1	55
Diciembre	CALDERON	Lunes	12-18	1	51
Diciembre	CALDERON	Lunes	18-00	1	52
Diciembre	CALDERON	Martes	00-06	1	5
Diciembre	CALDERON	Martes	06-12	1	33
Diciembre	CALDERON	Martes	12-18	1	28
Diciembre	CALDERON	Martes	18-00	1	62
Diciembre	CALDERON	Miercoles	00-06	1	19
Diciembre	CALDERON	Miercoles	06-12	1	20
Diciembre	CALDERON	Miercoles	12-18	1	28
Diciembre	CALDERON	Miercoles	18-00	1	46

3.4. Implementación del modelo SVM para la predicción del número de incidentes.

El aprendizaje de máquina del tipo SVM se puede realizar como regresión o clasificación, por lo que es indispensable aplicar los dos métodos de acuerdo al tipo de

datos de entrada y evaluar cada uno para saber cual tiene un mejor rendimiento; aunque para la tarea de predicción se puede decir que la mejor opción es la regresión, ya que no se necesita un valor exacto sino una estimación de un evento y que incluso puede tener un rango de error aceptable, que es justamente el concepto de regresión.

Sabiendo que es necesario aplicar funciones kernel cuando no se puede hacer una interpolación lineal de los valores de respuesta, se realiza una evaluación utilizando las tres el gaussiano, el polinómico y el lineal para determinar el de mejor rendimiento.

Sin embargo, de antemano se puede decir que la mejor opción es el kernel gaussiano porque es manejable principalmente a través de un par de hiperparámetros (C, σ), para poder mapear los datos en el nuevo espacio de características y separarlos del origen con el mayor margen posible (Reyna, 2009) para lograr una correcta clasificación, por lo tanto presenta baja complejidad contrario al kernel polinomial que usa más hiperparámetros; además su punto de operación es $0 < K_{ij} \leq 1$ mientras el polinomial puede llegar a infinito (Chih-Wei, Chih-Jen, & Chih-Chung, 2016). También se ha comprobado el rendimiento elevado del kernel gaussiano con respecto a los demás como se ve en (Gálvez, 2014).

Ya que el rendimiento del modelo SVM creado depende de los dos parámetros del kernel gaussiano (C, σ), es necesario realizar pruebas empíricas para saber cuál es el mejor par de estos y comparar el error que produce cada uno. Cabe mencionar que este par de hiperparámetros se utiliza para regresión y clasificación, pero también es importante el hiperparámetro ϵ en regresión y por lo tanto se debe incluir en las pruebas.

Teniendo listas las nuevas bases de datos de número de incidentes en las condiciones que requieren las funciones de Matlab, se puede aplicar el aprendizaje de máquinas con

la sintaxis respectiva; las bases de datos obtenidas después del procesamiento de los datos se guardan como tablas en una variable del tipo *table*, entonces si se guardó en la variable *base*, los ejemplos de las sintaxis para las cuatro bases de datos nuevas de número de incidentes son:

Mdl = fitrsvm(base, Num_Incidentes, 'KernelFunction', 'rbf', 'KernelScale', … 0.1, 'BoxConstraint', 2, 'Epsilon', 0.1, 'Standardize', true)

Mdl = fitcsvm(base, Num_Incidentes, 'KernelFunction', 'rbf', 'KernelScale', … 0.1, 'BoxConstraint', 2, 'Standardize', true)

Donde el par de argumentos *'KernelFunction', 'rbf'* especifican el kernel a utilizar que en este caso será el gaussiano radial (*'rbf'*), polinómico (*'polynomial'*) o lineal (*'linear'*), el par *'KernelScale', 0.1* determina el hiperparámetro σ para el kernel gaussiano, el par *'PolynomialOrder', 3* para determinar el grado del polinomio para el kernel polinómico, el par *'BoxConstraint', 2* determina el parámetro C de compensación para la maximización del margen entre valores de respuesta o clases el par *'Epsilon',0.1* de termina el hiperparámetro ϵ en la regresión y el par *'Standardize', true* sirven para estandarizar los valores predictores centrando y escalando cada columna o característica mediante la media ponderada de la columna y la desviación estándar respectivamente.

3.5. Implementación del modelo SVM para la predicción del tipo de incidentes.

La base de datos nueva con valores numéricos del tipo de incidentes, tiene los valores de respuesta en el rango entre 0 y 7 solamente, por lo tanto, no es necesario realizar cuantificación ni escalado, pero si se puede realizar codificación binaria para aplicar

clasificación de dos clases. Entonces se aplica las mismas sintaxis que para la base del número de incidentes de la subsección anterior para cada función de SVM en Matlab.

3.6. Predicción de datos usando las máquinas generadas.

Como se ve en las sintaxis de las subsecciones anteriores, los modelos de las SVM se guardan en la variable *Mdl*, entonces para utilizar dicho modelo para predecir la respuesta de un registro nuevo con las mismas características que los datos predictores, se utiliza la función *predict* donde la variable X es un vector con elementos tipo *simple* o *double*, que contiene las características del registro que se quiere predecir.

En el caso de la predicción para el modelo SVM de las base de datos con los valores de respuesta cuantificados uniformemente y con codificación binaria, se debe aplicar la función *predict* al mismo registro de predicción con cada uno de los modelos SVM dependiendo del número de dígitos que tenga el número binario de acuerdo a los niveles de cuantificación, siempre teniendo en cuenta el orden en que se aplican ya que cada modelo corresponde a un digito específico, desde el más significativo al menos significativo.

3.7. Procesamiento de datos de salida.

El resultado que muestra la función *predict* se basa en el modelo SVM utilizado como parámetro de entrada, es decir que el valor de respuesta que devuelve tiene el mismo formato que la base de datos que utilizo el modelo SVM para el aprendizaje, por lo que

se debe procesar de nuevo este resultado para tener el valor real del número de incidentes o el nombre del tipo de incidente más probable.

Entonces para tener el resultado en el formato original se debe realizar el proceso contrario al realizado en el tratamiento de los datos para cada uno de los modelos SVM de cada base de datos nueva de la siguiente manera:

- Modelo SVM del número de incidentes con datos numéricos: no es necesario ningún posprocesamiento del valor resultante.
- Modelo SVM del número de incidentes con datos numéricos cuantificados uniformemente: es necesario identificar el número correspondiente al reescalado desde el rango entre 0 y el número de niveles de cuantificación menos uno, al rango original entre 0 y 1570, lo cual se hace mediante un algoritmo en Matlab como una función (Anexo 8) que crea un vector con el rango de valores originales dividido en el número de niveles de cuantificación que se requiera y de acuerdo al valor cuantificado que se quiera decuantificar, se asigna a uno de los elementos de este vector por el número de su posición dentro del mismo.
- Modelo SVM del número de incidentes con datos numéricos cuantificados y escalados exponencialmente: en el caso de la construcción del modelo SVM es conveniente tener menor número de posibles niveles de respuesta para la predicción ya que la máquina se vuelve más precisa al no tener que redimensionar demasiado el espacio de características, en cambio al momento de utilizar el modelo ya para predecir es más conveniente aceptar el dato resultante sin tratar de escalarlo a niveles muy lejanos; entonces como son datos cuantificados igual al caso anterior, se utiliza el mismo código de Matlab para obtener el valor de respuesta original.

- Modelo SVM del número de incidentes con datos numéricos cuantificados uniformemente y con codificación binaria: primero se debe transformar el número binario resultante a decimal con la función de Matlab *binaryVectorToDecimal(binaryVector)* donde el único argumento de entrada es el vector binario que se quiere trasformar, y después aplicar el mismo procedimiento que a los datos cuantificados uniformemente.
- Modelo SVM del tipo de incidentes: sabiendo que el número de respuesta equivale a un tipo específico de incidente, se debe realizar la correspondencia de cada tipo de incidente, nombrado en una variable tipo *char*, con su respectivo número, en un script de Matlab como una función (Anexo 9) y luego elegir el nombre que corresponde a dicho número.

3.8. Creación de una interfaz gráfica en Matlab para el ingreso de datos y visualización de resultados.

Después de haber modelado las SVM con las diferentes tablas de la base de datos y determinado cuál de ellas tiene el mejor rendimiento en cuanto a la validez de la predicción del número de incidentes y el tipo, es decir el menor error, se puede guardar dicho modelo de máquina, en una archivo *.mat* con la siguiente línea de código:

save('Mdl.mat','Mdl');

Donde el primer argumento indica el nombre que tendrá del archivo *.mat* y el segundo el nombre de la variable que se quiere guardar; no necesariamente son iguales. Con el modelo ya guardado en un archivo es sencillo su manejo, entonces se comienza la

creación de la interfaz gráfica de usuario (GUI) mediante la herramienta GUIDE de Matlab, de la siguiente manera:

- Terminada la creación de la parte gráfica, se manda a correr dicha interfaz, entonces se guarda y se crean dos archivos, uno de ejecución de la ventana creada (*.fig*) ya apara su manejo y otro con el código de la interfaz (*.m*) para poder programar el funcionamiento de cada elemento de la ventana.
- Dentro del archivo de código se crean automáticamente funciones de llamado (Callback) y de propiedades (CreateFcn) de cada elemento de la interfaz gráfica; dentro de las funciones de llamado se declara lo que cada elemento debe hacer cuando se interactúa mediante la interfaz.

  En este caso se realiza un algoritmo en un Script nuevo (Anexo 10) para que al momento de presionar el botón “Predecir”, tome los valores de cada *Menú Emergente*, los ponga en un vector y lo utilice como el vector de valores de entrada para que la función *predict* realice la predicción utilizando el modelo SVM *Mdl* que se guardó como archivo. Obviamente la predicción se realiza una vez para el número de incidentes y otra para el tipo y con los modelos SVM correspondientes. Este algoritmo se ejecuta en la función de llamado del *Botón* (Anexo 11).
- Finalmente, al correr el archivo *.m* de la interfaz gráfica se abre la ventana de la nueva interfaz (Figura 6).

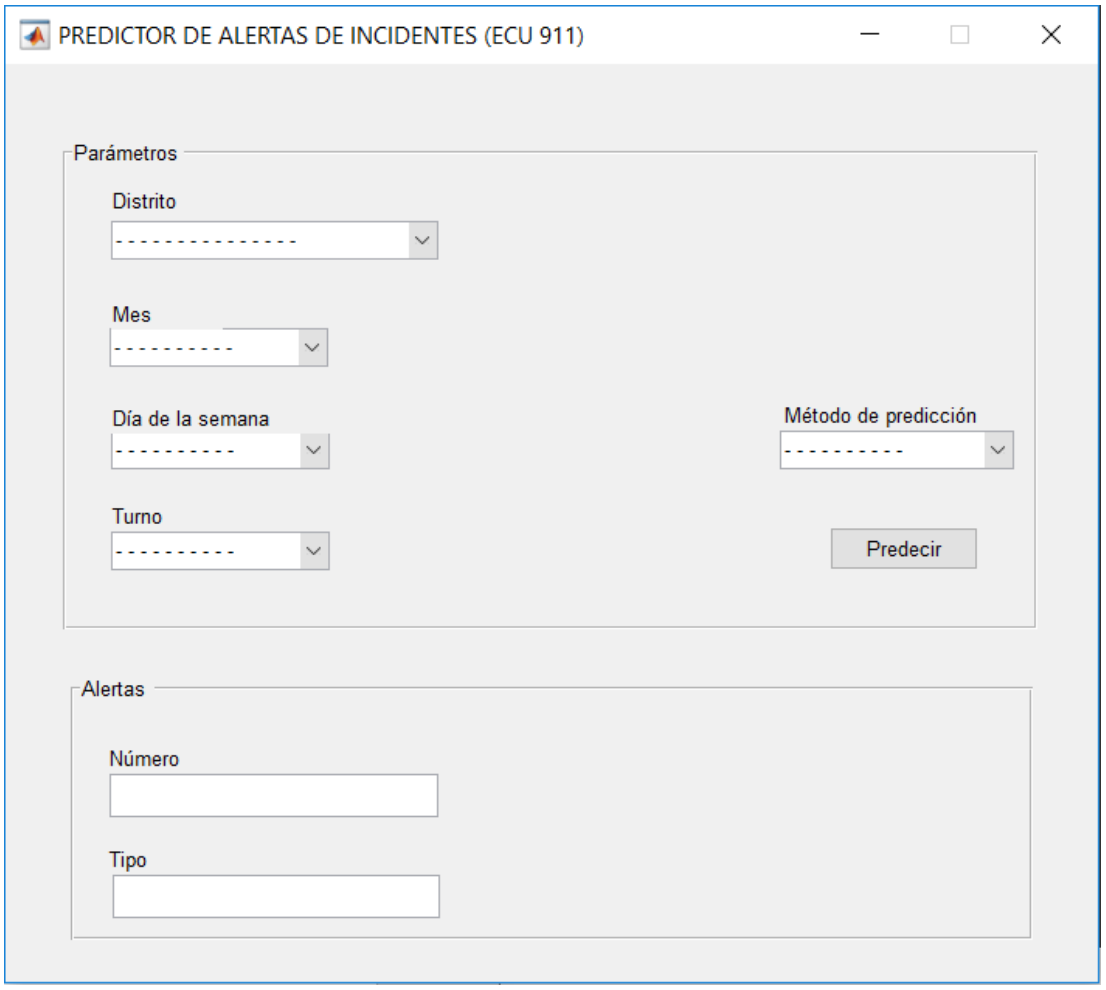

Figura 5 Interfaz gráfica

CAPÍTULO VI

4. RESULTADOS

La evaluación de los modelos SVM creados se hace aplicando la técnica k-fold con $k = 10$ en este caso, la cual cosiste en dividir la base de datos completa en k grupos con igual número de registros en cada uno y realizar el entrenamiento k veces con $k - 1$ grupos y el restante para probar el modelo generado, así se obtienen tantos registros de prueba como el tamaño de la base de datos, en este caso 24820.

Al obtener el error con cada una de las 24820 muestras predichas, y después de reescalar al nivel original cada una, se calculan dos errores diferentes como parámetros de evaluación, el primero es el error absoluto medio (MAE) que es un buen parámetro de evaluación mediante la fórmula (Giménez Fernández & Zamorano Cid, 2014):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Vp_i - Vr_i|$$

Siendo *Vp* el valor predicho, *Vr* el valor real y *n* el número de muestras que en este caso son 24820.

El segundo es el error relativo cuya fórmula es (Meneses Becerra, 2015):

$$ErrorRelativo = \frac{Vp - Vr}{Vr}$$

Sin embargo, en este caso se utiliza la fórmula alternativa:

$$ErrorRelativoMedio = \frac{Vp - Vr}{Vrm}$$

Donde Vrm es el promedio de los valores reales evaluados, ya que por el hecho de tener valores reales y predichos iguales a cero, no es posible realizar la división de la fórmula original que resultaría un valor infinito, por lo que se recomienda el uso de esta segunda fórmula.

4.1. Evaluación del Kernel y selección de hiperparámetros óptimos

4.1.1. Tratamiento de datos

Antes de realizar la evaluación del kernel e hiperparámetros se realiza una rápida evaluación del tratamiento de datos, es decir medir el rendimiento de modelos SVM con datos originales y normalizados; en este caso se usa como ejemplo el kernel gaussiano que por teoría tiene el mejor rendimiento y todas las bases de datos con respuestas procesadas.

Tabla 10

Error con datos originales y normalizados

		Niveles	MAE	Error relativo
Gaussiano	Original		30,752	0,538597879
	Cuantificado	64	31,39673016	0,539582202
		128	30,72552756	0,5501147
		256	30,614	0,530064838
		512	30,55318591	0,530239233
Gaussiano Normalizado	Original		31,48082755	0,560021627
	Cuantificado	64	31,60130159	0,542828718
		128	30,60886614	0,548989183
		256	30,66301961	0,531193658
		512	30,56032485	0,532993479
Gaussiano Normalizado (No respuestas)	Original		30,74999258	0,539572499
	Cuantificado	64	31,55146032	0,541748432
		128	30,62927559	0,548901481
		256	30,66482353	0,531605526
		512	30,57261448	0,530500101

Así se puede ver en la Tabla 9 donde se verifica, mediante resultados muy similares, que no es necesario realizar normalización debido a que las escalas de las características no tienen mucha diferencia y sabiendo que para regresión las respuestas no deben ser modificadas.

4.1.2. Kernel e hiperparámetros óptimos

El uso del kernel es la parte más importante en los casos no separables linealmente o ajustables por lo que es necesario validar la aplicación del mejor a través del análisis del error. Para poder discernir de manera más óptima y precisa se realizan pruebas con kernels gaussiano, polinomial y lineal para los modelos SVM del número y tipo de incidentes.

La selección de los hiperparámetros *C*, σ y ϵ óptimos, dependiendo si es regresión o clasificación, se realiza de forma empírica con cada una de las bases de datos nueva y cada función de modelado SVM que permite el ingreso de los mismos en Matlab.

Los rangos de valores para la prueba de cada hiperparámetro son: *C* = 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}, σ = 2^{-9}, 2^{-7}, 2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, ϵ = 2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5} y Grado = 2^{1}, 2^{3} entonces hacen todas las posibles combinaciones entre los tres y se tiene una tabla de resultados al final como el ejemplo de la Tabla 10 del kernel gaussiano para algunos *C* = 2^{-1} combinado con los σ y los ϵ, donde los resultados son el MAE y el promedio de los errores relativos como se menciona al inicio del capítulo.

Tabla 11

Pruebas para la selección de los hiperparámetros óptimos

Regresión			Clasificación		Numérico		Cuantif. 64	
C	σ	ε	C	σ	MAE	Error Relativo	MAE	Error Relativo
2^{-1}	2^{-9}	2^{-5}	2^{-1}	2^{-9}	37,6389303	0,869922301	23,61491053	0,545794929
2^{-1}	2^{-9}	2^{-3}	2^{-1}	2^{-7}	37,63549557	0,869842916	23,63888761	0,546349095
2^{-1}	2^{-9}	2^{-1}	2^{-1}	2^{-5}	37,63547542	0,869842451	28,24789212	0,652873796
2^{-1}	2^{-9}	2^{1}	2^{-1}	2^{-3}	37,64677679	0,870103652	42,45437179	0,981218236
2^{-1}	2^{-9}	2^{3}	2^{-1}	2^{-1}	37,73964545	0,87225006	160,7098353	3,714374148
2^{-1}	2^{-9}	2^{5}	2^{-1}	2^{1}	41,00398228	0,947696396	46,8883357	1,083697346
2^{-1}	2^{-7}	2^{-5}	2^{1}	2^{-9}	37,63855006	0,869913513	23,69766893	0,547707667
2^{-1}	2^{-7}	2^{-3}	2^{1}	2^{-7}	37,63269541	0,869778198	23,65637351	0,546753234
2^{-1}	2^{-7}	2^{-1}	2^{1}	2^{-5}	37,63781225	0,86989646	27,6126594	0,638192106
2^{-1}	2^{-7}	2^{1}	2^{1}	2^{-3}	37,64997985	0,870177682	42,46928296	0,981562868
2^{-1}	2^{-7}	2^{3}	2^{1}	2^{-1}	37,74031023	0,872265425	160,7088312	3,714350942
2^{-1}	2^{-7}	2^{5}	2^{1}	2^{1}	41,00569098	0,947735888	46,8883357	1,083697346
2^{-1}	2^{-5}	2^{-5}	2^{3}	2^{-9}	37,63397462	0,869807764	23,55985508	0,544522471
2^{-1}	2^{-5}	2^{-3}	2^{3}	2^{-7}	37,63604956	0,86985572	23,62015975	0,545916251
2^{-1}	2^{-5}	2^{-1}	2^{3}	2^{-5}	37,63753022	0,869889942	27,6380639	0,638779263
2^{-1}	2^{-5}	2^{1}	2^{3}	2^{-3}	37,64516519	0,870066404	42,44408567	0,9809805
2^{-1}	2^{-5}	2^{3}	2^{3}	2^{-1}	37,74194198	0,872303138	160,705819	3,714281324

CONTINÚA

2^{-1}	2^{-5}	2^{5}	2^{3}	2^{1}	41,0028783	0,947670881	46,8883357	1,083697346
2^{-1}	2^{-3}	2^{-5}	2^{5}	2^{-9}	37,62290688	0,869551963	23,54238837	0,544118775
2^{-1}	2^{-3}	2^{-3}	2^{5}	2^{-7}	37,61568335	0,869385011	23,56061612	0,544540061
2^{-1}	2^{-3}	2^{-1}	2^{5}	2^{-5}	37,62661928	0,869637765	26,28175243	0,607431783
2^{-1}	2^{-3}	2^{1}	2^{5}	2^{-3}	37,62839504	0,869678807	42,29344167	0,977498771
2^{-1}	2^{-3}	2^{3}	2^{5}	2^{-1}	37,72434662	0,871896469	160,7068231	3,71430453
2^{-1}	2^{-3}	2^{5}	2^{5}	2^{1}	40,9889425	0,947348793	46,8883357	1,083697346
2^{-1}	2^{-1}	2^{-5}	2^{7}	2^{-9}	35,44749533	0,819273196	25,11663341	0,580503201
2^{-1}	2^{-1}	2^{-3}	2^{7}	2^{-7}	35,45482967	0,819442709	24,89319353	0,575338991
2^{-1}	2^{-1}	2^{-1}	2^{7}	2^{-5}	35,44136945	0,819131612	26,25541166	0,606822987
2^{-1}	2^{-1}	2^{1}	2^{7}	2^{-3}	35,44969856	0,819324117	38,67668227	0,893907137
2^{-1}	2^{-1}	2^{3}	2^{7}	2^{-1}	35,62400825	0,823352815	160,7089016	3,714352568
2^{-1}	2^{-1}	2^{5}	2^{7}	2^{1}	39,74908556	0,918692845	46,8883357	1,083697346
2^{-1}	2^{1}	2^{-5}	2^{9}	2^{-9}	36,45203147	0,842490338	34,82693488	0,804930615
2^{-1}	2^{1}	2^{-3}	2^{9}	2^{-7}	36,46529815	0,842796961	34,72779696	0,80263931
2^{-1}	2^{1}	2^{-1}	2^{9}	2^{-5}	36,45545952	0,842569568	35,208404	0,813747245
2^{-1}	2^{1}	2^{1}	2^{9}	2^{-3}	36,48282658	0,843202083	41,64307778	0,962467364
2^{-1}	2^{1}	2^{3}	2^{9}	2^{-1}	36,67160065	0,847565086	160,7202391	3,714614604
2^{-1}	2^{1}	2^{5}	2^{9}	2^{1}	40,10919872	0,927015889	46,8883357	1,083697346

A continuación, se muestran solamente los resultados de la combinación de hiperparámetros más óptima.

4.2. Evaluación de las máquinas creadas del número de incidentes

A continuación, se presentan los resultados del MAE y error relativo promedio con los hiperparámetros óptimos de cada posible tratamiento de las respuestas de la base de datos, para el número de incidentes y con los kernel gaussiano (Tabla 11), polinómico (Tabla 12) y lineal (Tabla 13).

Tabla 12

Hiperparámetros óptimos kernel gaussiano (número de incidentes)

Kernel Gaussiano					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$C = 2^{9}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-3}$	14,71291917	0,340049422
	Cuantificación uniforme	64	$C = 2^{7}$, $\sigma = 2^{-3}$ y $\epsilon = 2^{-3}$	15,90928143	0,367700107
		128	$C = 2^{5}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{-3}$	15,06874409	0,348273355
		256	$C = 2^{5}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-5}$	14,78258915	0,341659656
		512	$C = 2^{9}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-5}$	14,74999842	0,340906409
	Cuantificación con escala exponencial	64	$C = 2^{5}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{-1}$	14,66174531	0,338866676
		128	$C = 2^{7}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{1}$	14,47764179	0,33461162
		256	$C = 2^{7}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{1}$	14,56628938	0,33666047
		512	$C = 2^{9}$, $\sigma = 2^{-3}$ y $\epsilon = 2^{3}$	14,4468119	0,333899071
	Cuantificación uniforme y codificación binaria	64	$C = 2^{9}$ y $\sigma = 2^{-1}$	19,23310119	0,444521231
		128	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,51250167	0,427866518
		256	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,35444613	0,424213491
		512	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,06657681	0,417560168

Tabla 13

Hiperparámetros óptimos kernel polinómico (número de incidentes)

Kernel Polinómico					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$G = 3$, $C = 2^{5}$ y $\epsilon = 2^{-5}$	36,05419303	0,833295376
	Cuantificación uniforme	64	$G = 3$, $C = 2^{1}$ y $\epsilon = 2^{-3}$	36,67531497	0,847650933
		128	$G = 3$, $C = 2^{-1}$ y $\epsilon = 2^{-3}$	35,96307302	0,831189383
		256	$G = 3$, $C = 2^{3}$ y $\epsilon = 2^{-3}$	35,93629821	0,830570555
		512	$G = 3$, $C = 2^{-1}$ y $\epsilon = 2^{-3}$	35,92201092	0,830240343
	Cuantificación con escala exponencial	64	$G = 3$, $C = 2^{1}$ y $\epsilon = 2^{-1}$	32,95531803	0,761673243
		128	$G = 3$, $C = 2^{5}$ y $\epsilon = 2^{1}$	33,31394269	0,769961884
		256	$G = 3$, $C = 2^{3}$ y $\epsilon = 2^{-5}$	33,34528158	0,770686198
		512	$G = 3$, $C = 2^{7}$ y $\epsilon = 2^{3}$	33,32795536	0,770285749
	Cuantificación uniforme y codificación binaria	64	$G = 3$ y $C = 2^{5}$	42,42168054	0,980462666
		128	$G = 3$ y $C = 2^{5}$	42,80563141	0,989336654
		256	$G = 3$ y $C = 2^{5}$	42,91501872	0,991864847
		512	$G = 3$ y $C = 2^{5}$	42,29622669	0,977563139

Tabla 14

Hiperparámetros óptimos kernel lineal (número de incidentes)

Kernel Lineal					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$C = 2^{-1}$ y $\epsilon = 2^{-3}$	38,10598419	0,880716991
	Cuantificación uniforme	64	$C = 2^{1}$ y $\epsilon = 2^{-5}$	38,86034496	0,898152005
		128	$C = 2^{5}$ y $\epsilon = 2^{-1}$	38,25087623	0,88406578
		256	$C = 2^{3}$ y $\epsilon = 2^{-5}$	38,13662211	0,881425104
		512	$C = 2^{9}$ y $\epsilon = 2^{-3}$	38,11717343	0,8809756
	Cuantificación con escala exponencial	64	$C = 2^{5}$ y $\epsilon = 2^{1}$	38,14255399	0,881562204
		128	$C = 2^{1}$ y $\epsilon = 2^{-5}$	38,14150368	0,881537928
		256	$C = 2^{-1}$ y $\epsilon = 2^{-1}$	38,14330246	0,881579502
		512	$C = 2^{5}$ y $\epsilon = 2^{5}$	38,13758408	0,881447337
	Cuantificación uniforme y codificación binaria	64	$C = 2^{-1,\,9}$	43,26700242	1
		128	$C = 2^{-1,\,9}$	43,26700242	1
		256	$C = 2^{-1,\,9}$	43,26700242	1
		512	$C = 2^{-1,\,9}$	43,26700242	1

Para poder comparar y determinar el mejor modelo SVM para trabajar en la interfaz gráfica para el usuario final se tienen los resultados de errores presentados en las tablas anteriores y las gráficas: una que superpone una curva del número de cada una de las pruebas en el eje de las abscisas vs el valor de respuesta original en el eje de las ordenadas y una curva con el mismo eje de abscisas vs el valor de respuesta predicho en el eje de las ordenadas, para poder compararlas, otra gráfica igual pero ordenada de mayor a menor según el valor de respuesta y una final con los errores relativos de las pruebas.

Ya que en las tablas se ve claramente el mejor rendimiento de los modelos SVM con kernel gaussiano, en las Figuras 7, 8 y 9 se pueden ver las gráficas mencionadas en el mismo orden que se mencionaron en el párrafo anterior y en dos partes, la figura a con

las 24820 pruebas realizadas y la b con las primeras 1000 muestras, de los resultados obtenidos con los modelos SVM de la base de datos cuantificada a escala exponencial con 512 niveles con kernel gaussiano como ejemplo por ser el de mejor rendimiento.

La primera muestra un patrón muy parecido en las dos curvas con errores notables en los valores más altos sobre los 1200 incidentes, la segunda con los datos ordenados se puede apreciar de mejor manera que existe un mayor rango de errores en la predicción, pero hay que notar que los más grandes son a partir de los 600 incidentes y que no son más de 25 muestras; para la tercera grafica con los errores relativos ordenados también en forma ascendente, se notan picos justamente en las muestras de las respuestas de mayor número de incidentes.

Estos errores grandes para los valores altos de número de incidentes se dan porque existen muy pocos registros con respuestas mayores a 600 incidentes, entonces el aprendizaje en ese rango es más complejo y el modelo SVM no es capaz de una buena generalización al predecir. Sin embargo, se puede decir que el rendimiento del modelo SVM es bueno y es factible para predicción de datos.

Para el resto de modelos de predicción de número de incidentes, no es necesario especificar las gráficas de comparación porque tiene un comportamiento similar a las vistas anteriormente en cada uno de los niveles de cuantificación, con cambios mínimos.

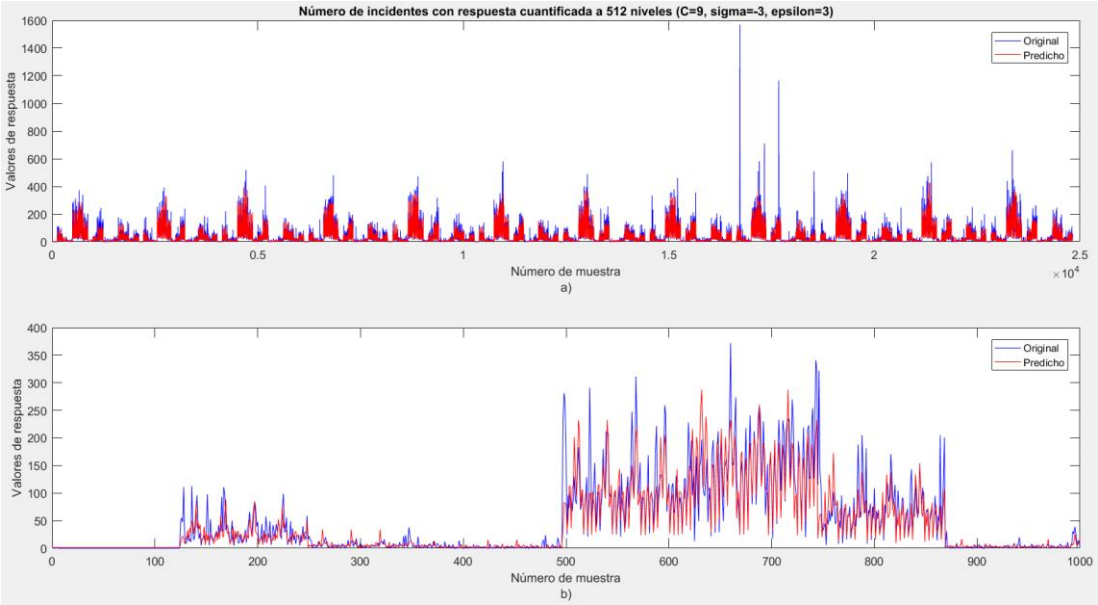

Figura 6 Respuesta cuantificada uniformemente (512 niveles-por muestra)

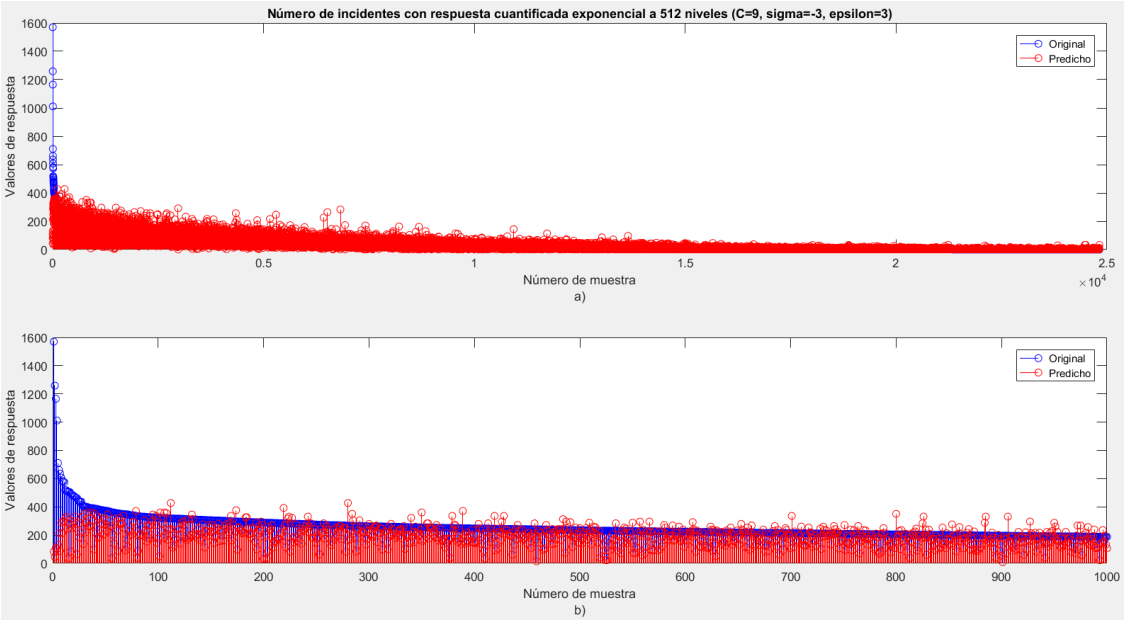

Figura 7 Respuesta cuantificada uniformemente (512 niveles-por respuesta)

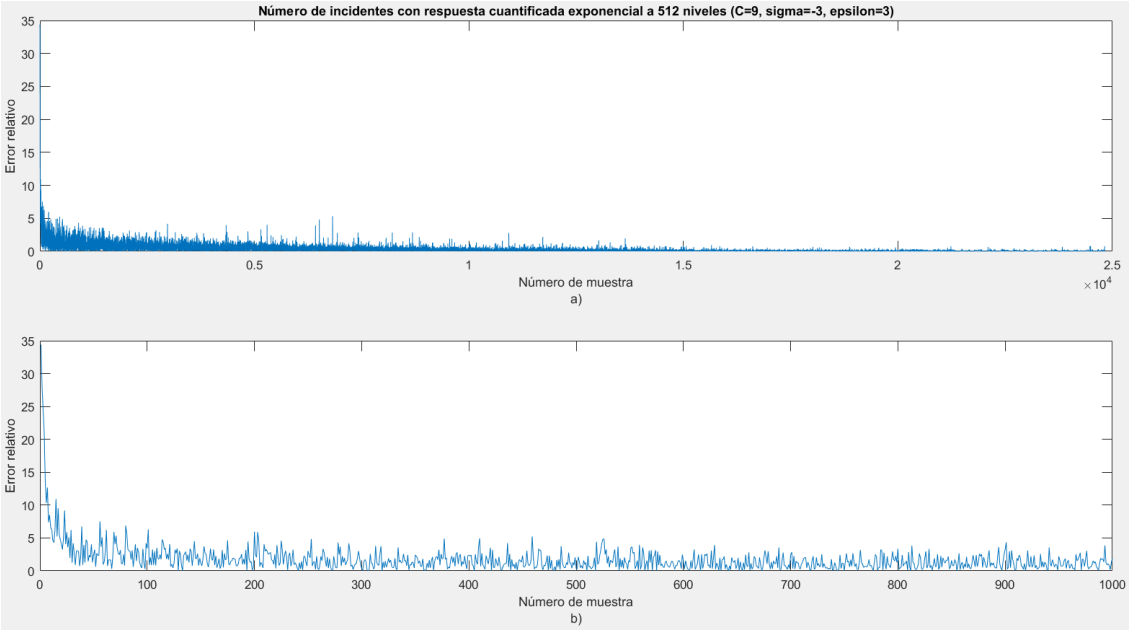

Figura 8 Respuesta cuantificada uniformemente

(512 niveles-error relativo por respuesta)

4.3. Evaluación de las máquinas creadas según los tipos de incidentes

Al igual que la sección anterior, en la Tabla 14, Tabla 15 y Tabla 16 se tienen los resultados del MAE y error relativo promedio de los modelos SVM de las bases de datos con todos los posibles valores de respuesta procesados para poder comparar y determinar el mejor modelo SVM para la predicción de los tipos de incidentes para trabajar en la interfaz gráfica para el usuario final.

También se puede ver las Figuras 9, 10 y 11 que comparan los datos originales con los predichos en el mismo orden que las del número de incidentes y en dos partes, la figura a con las 24820 pruebas realizadas y la b, para las Figura 9 y Figura 11 la comparación solamente de las ultimas 1000 muestras para visualizar mejor la diferencia

y para la Figura 10 solamente se grafican los datos originales para comparar mejor con los predichos q se sobre ponen en la figura a; esto del modelo SVM de respuestas con codificación binaria con kernel gaussiano, con un comportamiento totalmente diferente a los modelos de predicción del número de incidentes.

A pesar de tener valores similares en los errores del kernel gussiano y polinómico, se elige como mejor al gaussiano debido a que en las demás pruebas de los hiperparámetros se tienen mejores resultados para este último.

Tabla 15

Hiperparámetros óptimos kernel gaussiano (tipo de incidentes)

Kernel Gaussiano					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Tipo	Original		$C = 2^3$, $\sigma = 2^{-1}$ y $\epsilon = 2^{-5}$	0,990687709	0,265240647
	Codificación binaria		$C = 2^3$ y $\sigma = 2^1$	0,922119259	0,246882551

Tabla 16

Hiperparámetros óptimos kernel polinómico (tipo de incidentes)

Kernel Polinómico					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Tipo	Original		$G = 3$, $C = 2^{-1}$ y $\epsilon = 2^{-5}$	1,130792721	0,302751503
	Codificación binaria		$G = 3$ y $C = 2^{-1}$	0,921998388	0,24685019

Tabla 17

Hiperparámetros óptimos kernel lineal (tipo de incidentes)

Kernel Lineal					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Tipo	Original		$C = 2^{3}$ y $\epsilon = 2^{-5}$	1,131761974	0,303011005
	Codificación binaria		$C = 2^{-1,\,9}$	1,123609992	0,300828443

En la Figura 9 no es muy notoria la diferencia entre las dos curvas ya que la mayoría de los datos se sobre ponen, pero en la Figura 10 se puede ver que solo los valores más altos y más bajos presentan errores notables que son menos de la tercera parte del total de pruebas y lo cual se da, al igual que el caso del número de incidentes, por falta de registros con esos valores de respuesta y por lo tanto no hay buena generalización del modelo SVM; así mismo los errores relativos de la Figura 11 muestra picos en los valores más altos coincidiendo con la Figuras 10, por lo que se puede verificar un rendimiento óptimo del modelo SVM.

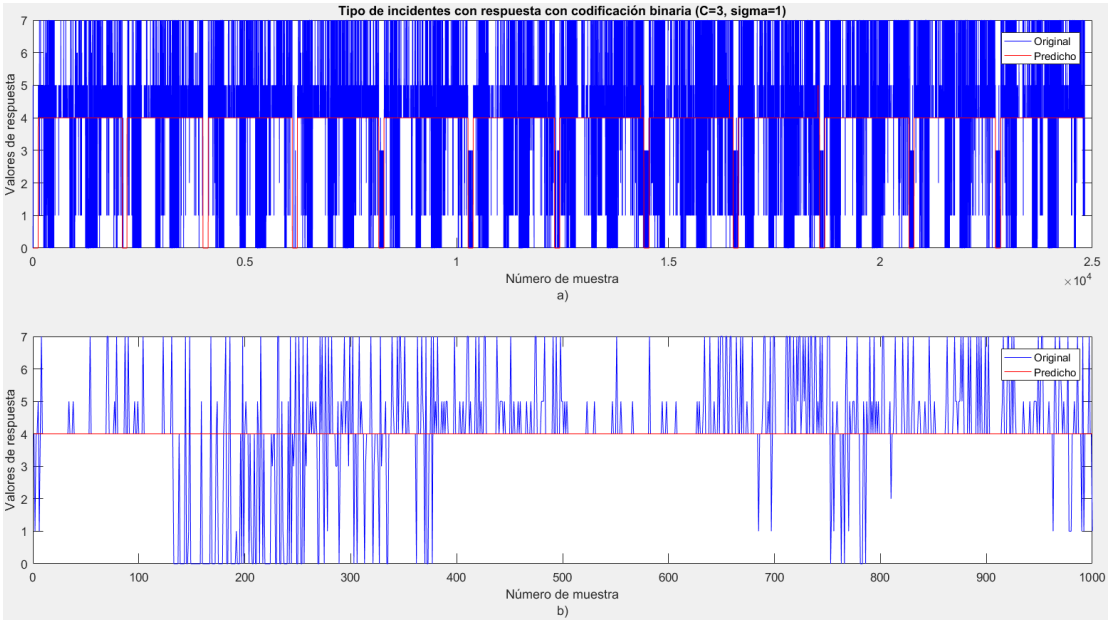

Figura 9 Respuesta numérica original, tipo de incidentes (por muestra)

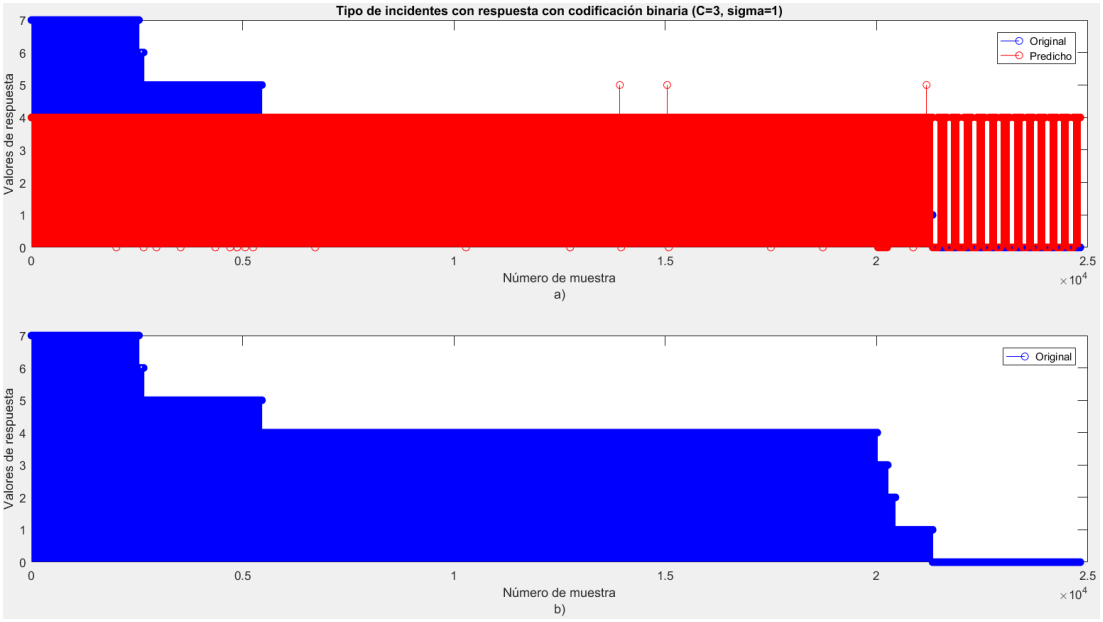

Figura 10 Respuesta numérica original, tipo de incidentes (por respuesta)

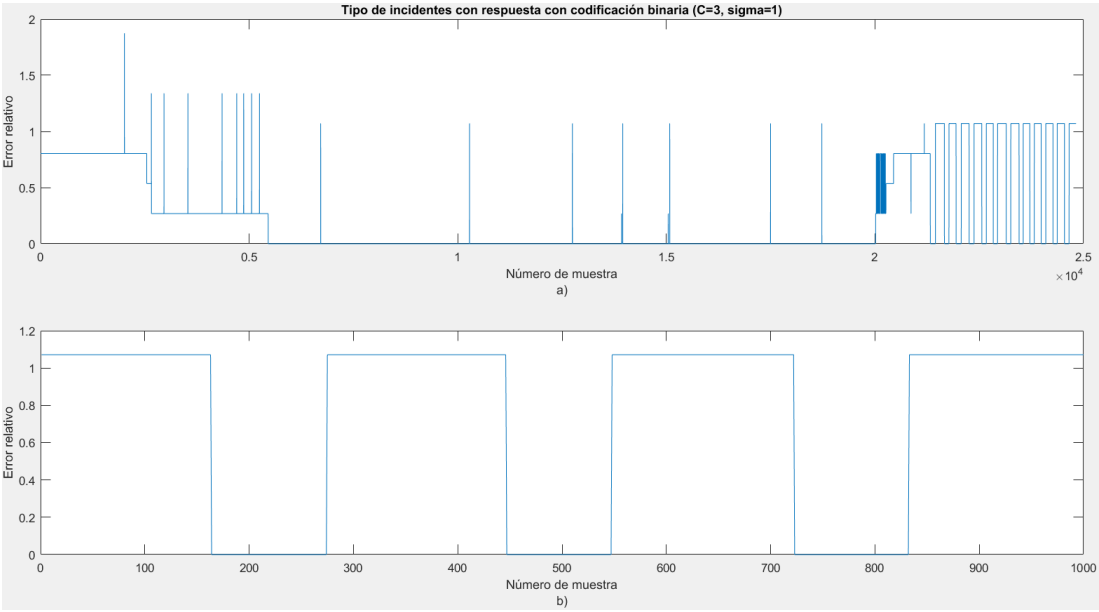

Figura 11 Respuesta numérica original, tipo de incidentes

(error relativo por respuesta)

4.4. Evaluación con característica complementaria

Con la característica adicional del número del día de la semana dentro del mes, se obtienen los siguientes resultados mostrados en la Tabla 17, Tabla 18 y Tabla 19 para el número de incidentes y en la Tabla 20 y Tabla 21 para el tipo de incidentes.

Tabla 18

Hiperparámetros óptimos kernel gaussiano (número de incidentes)

Kernel Gaussiano					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$C = 2^{9}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-3}$	25,52586692	0,589961529
	Cuantificación uniforme	64	$C = 2^{7}$, $\sigma = 2^{-3}$ y $\epsilon = 2^{-3}$	25,02977565	0,578495719
		128	$C = 2^{5}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{-3}$	23,85412926	0,551323825
		256	$C = 2^{5}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-5}$	23,9140881	0,552709612
		512	$C = 2^{9}$, $\sigma = 2^{-7}$ y $\epsilon = 2^{-5}$	23,61875744	0,54588384
	Cuantificación con escala exponencial	64	$C = 2^{5}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{-1}$	27,22777196	0,629296472
		128	$C = 2^{7}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{1}$	26,30400759	0,607946151
		256	$C = 2^{7}$, $\sigma = 2^{-9}$ y $\epsilon = 2^{1}$	27,20934554	0,628870595
		512	$C = 2^{9}$, $\sigma = 2^{-3}$ y $\epsilon = 2^{3}$	26,21126793	0,600425184
	Cuantificación uniforme y codificación binaria	64	$C = 2^{9}$ y $\sigma = 2^{-1}$	19,2103456	0,443995297
		128	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,7069061	0,432359652
		256	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,24756837	0,4217433
		512	$C = 2^{9}$ y $\sigma = 2^{-1}$	18,01327124	0,419440455

Tabla 19

Hiperparámetros óptimos kernel polinómico (número de incidentes)

Kernel Polinómico					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$G = 3$, $C = 2^{5}$ y $\epsilon = 2^{-5}$	36,07700676	0,833822653
	Cuantificación uniforme	64	$G = 3$, $C = 2^{1}$ y $\epsilon = 2^{-3}$	36,67136782	0,847559705
		128	$G = 3$, $C = 2^{-1}$ y $\epsilon = 2^{-3}$	35,98615195	0,83172279
		256	$G = 3$, $C = 2^{3}$ y $\epsilon = 2^{-3}$	35,95242197	0,830867704
		512	$G = 3$, $C = 2^{-1}$ y $\epsilon = 2^{-3}$	35,94532984	0,830814912
	Cuantificación con escala exponencial	64	$G = 3$, $C = 2^{1}$ y $\epsilon = 2^{-1}$	32,98871354	0,76244509
		128	$G = 3$, $C = 2^{5}$ y $\epsilon = 2^{1}$	33,34320832	0,77063828
		256	$G = 3$, $C = 2^{3}$ y $\epsilon = 2^{-5}$	33,35490152	0,770908537
		512	$G = 3$, $C = 2^{7}$ y $\epsilon = 2^{3}$	33,33917173	0,770807231
	Cuantificación uniforme y codificación binaria	64	$G = 3$ y $C = 2^{5}$	42,42168054	0,980462666
		128	$G = 3$ y $C = 2^{5}$	42,80563141	42,80563141
		256	$G = 3$ y $C = 2^{5}$	42,91501872	0,991864847
		512	$G = 3$ y $C = 2^{5}$	42,29622682	0,977563139

Tabla 20

Hiperparámetros óptimos kernel lineal (número de incidentes)

Kernel Lineal					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Número	Original		$C = 2^{-1}$ y $\epsilon = 2^{-3}$	38,10277771	0,88002962
	Cuantificación uniforme	64	$C = 2^{-1}$ y $\epsilon = 2^{-5}$	38,88031477	0,898613553
		128	$C = 2^{5}$ y $\epsilon = 2^{-5}$	38,27499952	0,884623325
		256	$C = 2^{3}$ y $\epsilon = 2^{-5}$	38,13671647	0,881508408
		512	$C = 2^{1}$ y $\epsilon = 2^{-3}$	38,11957736	0,880955653
	Cuantificación con escala exponencial	64	$C = 2^{-1}$ y $\epsilon = 2^{1}$	38,14436728	0,881535236
		128	$C = 2^{-1}$ y $\epsilon = 2^{-1}$	38,14560283	0,881632669
		256	$C = 2^{-1}$ y $\epsilon = 2^{-1}$	38,1410217	0,881526789
		512	$C = 2^{-1}$ y $\epsilon = 2^{-1}$	38,14180976	0,881545003
	Cuantificación uniforme y codificación binaria	64	$C = 2^{-1,9}$	43,26700242	1
		128	$C = 2^{-1,9}$	43,26700242	1
		256	$C = 2^{-1,9}$	43,26700242	1
		512	$C = 2^{-1,9}$	43,26700242	1

Tabla 21

Hiperparámetros óptimos kernel gaussiano (tipo de incidentes)

Kernel Gaussiano					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Tipo	Original		$C = 2^{3}$, $\sigma = 2^{-1}$ y $\epsilon = 2^{-5}$	1,016018756	0,272022626
	Codificación binaria		$C = 2^{3}$ y $\sigma = 2^{1}$	0,95470854	0,257026514

Tabla 22

Hiperparámetros óptimos kernel polinómico (tipo de incidentes)

Kernel Polinómico					
Incidentes	**Respuesta**	**Niveles**	**Hiperparámetros óptimos**	**MAE**	**ERM**
Tipo	Original		G = 3, $C = 2^{-1}$ y $\epsilon = 2^{-5}$	1,156002103	0,314306311
	Codificación binaria		G = 3 y $C = 2^{-1}$	0,952141445	0,261464777

Tabla 23

Hiperparámetros óptimos kernel lineal (tipo de incidentes)

Kernel Lineal					
Incidentes	Respuesta	Niveles	Hiperparámetros óptimos	MAE	ERM
Tipo	Original		$C = 2^{3}$ y $\epsilon = 2^{-3}$	1,140664387	0,309875023
	Codificación binaria		$C = 2^{-1,\,9}$	1,123609992	0,300828443

Los resultados de estas tablas reflejan que la característica extra no es relevante para el entrenamiento del modelo SVM, ya que al no aporta información importante a la combinación de las demás características y las etiquetas de cada registro, así al momento de predecir las muestras de prueba, había menos coincidencia con esta nueva característica y por eso aumenta el MAE y el error relativo promedio.

Además de dichos resultados numéricos, se puede ver en la base de datos que los resultados de cada registro son similares dentro de todo un mes en cada distrito, excepto en meses críticos como enero y diciembre en los cuales si influye esta nueva característica pero que tampoco es muy relevante en el modelado SVM por representar solo el 16.7% de los datos totales.

Entonces se concluye de esta sección que los modelos SVM creados sin dicha característica complementaria son los más acertados y con los que se trabaja para conseguir el modelo final para la interfaz gráfica.

4.5. Selección del modelo SVM final

En los resultados numéricos se observa claramente que los mejores modelos son los entrenados con kernel gaussiano, así, para el caso de los modelos SVM para la

predicción del número de incidentes, se pueden descartar aquellos basados en los valores de respuesta con codificación binaria porque su rendimiento es menor y muy diferente a los otros tres con bases de datos con respuestas originales, cuantificadas y cuantificadas a escala exponencial, los cuales tienen una diferencia del MAE de máximo un punto y del error relativo promedio tres puntos porcentuales; entonces se realiza una comparación visual entre estos dos últimos como se ve en la Figura 13.

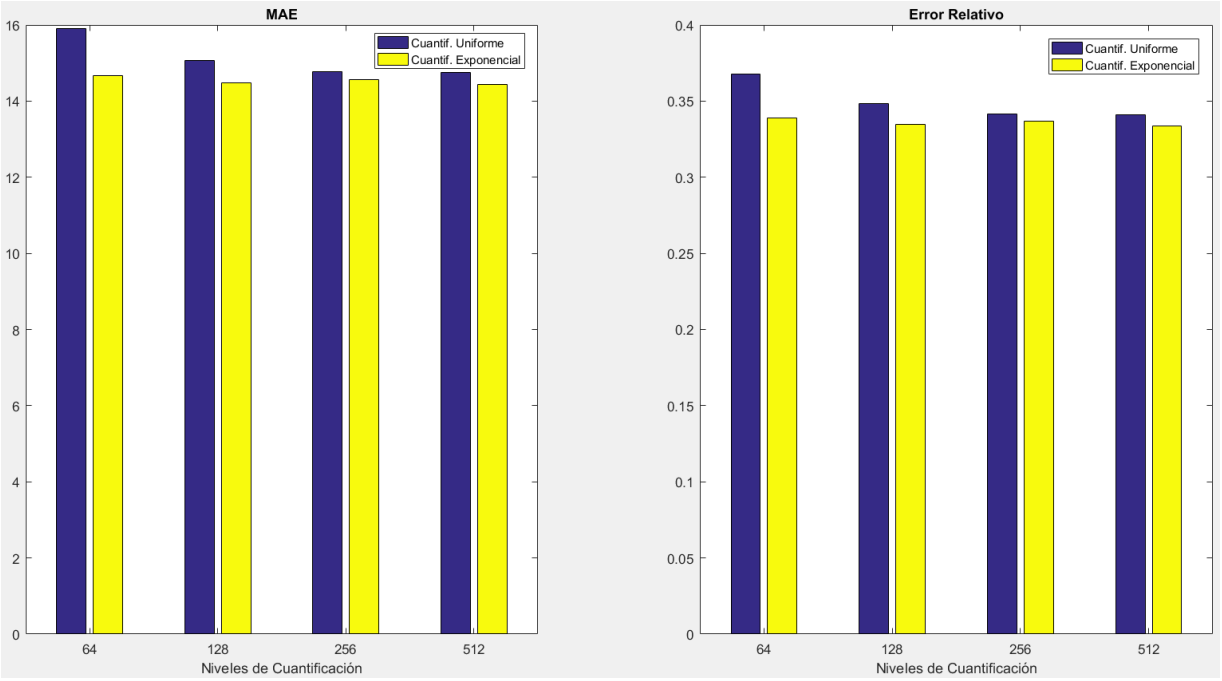

Figura 12 Histograma de comparación de modelos SVM cuantificados y con escala exponencial

De estos resultados obtenidos y a pesar de la mínima diferencia en los resultados de los errores, se verifica que los modelos SVM del número de incidentes de mejor rendimiento son: el de las respuestas cuantificadas a escala exponencial a 128 y 512

niveles, por lo que se escoge el segundo que se acerca más al máximo número de incidentes original.

Sin embargo, para la interfaz gráfica se eligen como métodos de predicción los siguientes modelos SVM: Método 1 el de las respuestas cuantificadas a escala exponencial a 512 niveles escogido por los mejores resultados de errores, Método 2 el de las respuestas cuantificadas uniformemente a 512 niveles por ser el mejor de todos los cuantificados uniformemente y para no perder los niveles que el primero descarta cuando se ingresen datos nuevos justamente en dichos niveles, y Método 3 el de respuestas cuantificadas a escala exponencial a 128 niveles por ser el siguiente con mejor rendimiento.

Se esperaban estos resultados ya que, mediante la cuantificación, el modelo SVM tiene menos posibilidades de selección de respuesta y por lo tanto menos posibilidades de equivocarse; así mismo tampoco podía ser más eficiente el de menor número de niveles de cuantificación (64 niveles) porque se aleja más de los valores originales y al final debe compararse y regresar a estos valores.

En cuanto al modelo SVM del tipo de incidentes, se verifica que el de respuestas con codificación binaria es el mejor, ya que la predicción se vuelve más exacta cuando se realiza clasificación binaria y se combinan pocos modelos SVM, en este caso solo tres por tener ocho respuestas posibles.

El paso final es crear los modelos SVM con los datos procesados para los tres métodos como se mencionó para la predicción del número de incidentes y codificación binaria para la predicción del tipo de incidentes, con los hiperparámetros correspondientes, para guardarlos en las variables *Mdl1*, *Mdl*2, *Mdl3* los del número de

incidentes y *Mdl4, Mdl5 y Mdl6* los del tipo de incidentes respectivamente y poder usarlos en la predicción de la interfaz gráfica.

CAPÍTULO VII

5. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones.

Cuando se procesan grandes cantidades de datos, como en este caso más de 50000 registros, es necesario realizar algoritmos que automaticen dicho proceso, ya que es posible tener que realizar las mismas pruebas varias veces debido al cambio que sufren los mismos datos o su tratamiento de acuerdo al resultado que se quiere obtener.

De todas las características disponibles en la base de datos original era necesario elegir las más relevantes, primero para la creación de los nuevos registros del conteo del número y tipo de incidentes y segundo para la etiqueta que se quiere como respuesta de cada uno de estos nuevos registros, en este caso también fue necesario tomar en cuenta que dichas características son también los datos que va a ingresar el usuario mediante la interfaz gráfica y por lo tanto debían ser sencillas y lógicas para consultar el número y tipo de incidentes.

Debido a la complejidad que presenta la creación de hiperplanos con funciones polinómicas en *n* dimensiones para lograr la separación (clasificación) o ajuste (regresión) lineal de los registros de la base de datos, se obtiene un error mayor al usar este kernel en comparación al gaussiano, como se ve en las Tabla 11, Tabla 12 y Tabla

13, además de su manejabilidad mediante los hiperparámetros para obtener un balance entre la generalización y la precisión del modelo SVM.

Sabiendo que cuando se aplica regresión, no se debe normalizar los datos, especialmente las respuestas, se comprueba en la Tabla 9 que de esta manera las respuestas tienen mayor error al momento de predecir usando el modelo normalizado, debido a la cercanía numérica entre cada etiqueta, la discriminación es más difícil de realizar.

Se realizó un filtrado de datos, en cuanto al número de registros, correspondientes al centro de operaciones Quito debido a que la gran cantidad existente de estos no permitía el correcto desempeño del proceso de aprendizaje del modelo SVM y por lo tanto final con datos nuevos, principalmente por la cantidad de etiquetas de valor cero que no correspondían a este centro de operaciones. Se redujo la base por lo menos al 60% del total.

Se optó por utilizar la técnica de evaluación K-fold para la selección del mejor procesamiento de datos (entre varios niveles de cuantificación y codificación) ya que se sabe que al tener una mayor cantidad de datos de aprendizaje se consigue una mejor generalización del modelo de aprendizaje y por eso con este método se obtienen 24820 registros de prueba, es decir el mismo número de registros de la base original, disminuyendo así la probabilidad de error.

El criterio para elegir el mejor procesamiento de los datos de aprendizaje se basó en una combinación entre el MAE y el error relativo, el MAE resulta de una diferencia absoluta entre el valor predicho y el original, así se puede determinar el modelo que mejor rinde al momento de las pruebas tomándolo como una desviación estándar o rango

permitido de error; además con el criterio de no perder una cantidad significativa de datos, como es el caso de la cuantificación a bajo número de niveles para el modelo de número de incidentes.

Los mejores resultados se obtienen aplicando el aprendizaje de máquinas de regresión SVM completo, es decir, modificando todos los hiperparámetros disponibles dentro de la aplicación del kernel, como se ve en los ejemplos de la Tabla 10, para tener al final un equilibrio justo entre la capacidad de generalización y el mínimo error del modelo SVM.

La aplicación de codificación binaria en las respuestas o etiquetas, resulta eficiente cuando estas son mínimas, como el caso del modelo SVM del número de incidentes donde solo hay ocho niveles posibles.

5.2. Recomendaciones y trabajos futuros.

Se debe automatizar cualquier proceso realizado a una base de datos mediante la construcción de algoritmos, especialmente si es muy extensa, para así facilitar cualquier tipo de prueba que se realice en la investigación.

Es recomendable siempre visualizar los datos mediante gráficas, ya que solo así es posible apreciar la magnitud de los valores resultantes como pueden ser, en este caso, las respuestas del modelo SVM, los errores, las diferencias entre predicciones y valores originales, etc.

Para acelerar el procesamiento de datos y el aprendizaje de máquina es recomendable utilizar una computadora que tenga buen procesador de mínimo 2.5GHz y core i5 y con una memoria RAM de por lo menos 8 GB.

Como trabajo futuro se debe aumentar las bases de datos de los incidentes registrados en los años siguientes, 2015 y 2016, en el proceso de aprendizaje de máquina y así mejorar la exactitud, precisión y capacidad de generalización de la predicción con el modelo SVM final.

REFERENCIAS

Carmona Suárez, E. (2014). Tutorial sobre Máquinas de Vectores Soporte (SVM). ResearchGate: Madrid, España. Recuperado de https://www.researchgate.net/publication/263817587_Tutorial_sobre_Maquinas_de_Vectores_Soporte_SVM

Carrera Erazo, E. V. (2016). *Machine Learning for Signal Processing.* Universidad de las Fuerzas Armadas ESPE, Eléctrica y Electrónica, Quito.

Hsu, C. W., Chang, C. C., y Lin, C. J. (2003). A Practical Guide to Support Vector Classification. *Data Science Association.*

ECU 911. (2016). La institución. Servicio Integrado de Seguridad ECU 911. Recuperado de http://www.ecu911.gob.ec/la-institucion/

ECU-911 Quito. (2013). Emergencias atendidas por ECU911 Quito. 25/11/2013. *Infogram: Ecuador*. Recuperado de https://infogr.am/emergencias-atendidas-por-ecu911-quito-25112013

Secretaría Nacional de Comunicación. (2014). ECU 911 cuenta con 15 centros operativos para dar cobertura a todo el territorio ecuatoriano. *El Ciudadano-Sistema de información oficial*: Ecuador. Recuperado de http://www.elciudadano.gob.ec/ecu-911-cuenta-con-15-centros-operativos-para-dar-cobertura-a-todo-el-territorio-ecuatoriano/

Gala García, Y. (2013). *Algoritmos SVM para problemas sobre big data* (tesis pregrado). Universidad Autónoma de Madrid, Madrid, España.

Gálvez, S. (2014). Detector De Dígitos Con SVM. Universidad de Chile, Departamento de Ingeniería Eléctrica, Santiago de Chile.

Giménez, R., Zamorano, P. (2014). Modelos Predictivos de Índices Bursátiles Relevantes para la Economía Chilena (tesis de pregrado). Universidad de Chile, Santiago, Chile.

Matlab R2016a. (2016). Matlab R2016a Documentation.

Mandle, A. K., Jain, P., y Shrivastava, S. K. (2012). Protein Structure Prediction Using Support Vector Machine. (02 de 2012). *International Journal on Soft Computing (IJSC), 3*(1), 67.

Meneses, F. (2015). Métodos Numéricos En Los Lenguajes Matlab Y Microsoft Visual C#.Net, Quito, Ecuador: Comisión editorial de la Universidad de las Fuerzas Armadas ESPE.

Radhika, Y., & Shashi, M. (2009). Atmospheric Temperature Prediction using Support Vector Machines. *International Journal of Computer Theory and Engineering, 1*(1), 55.

Reyna, A. (02 de 2009). SVM de una clase: aplicación a detección de novedad. Universidad de la República, Uruguay.

Sancho Caparrini, F. (2017). Introducción al Aprendizaje Automático. Sevilla, España: *Universidad de Sevilla - Fernando Sancho Caparrini*. Recuperado de http://www.cs.us.es/~fsancho/?e=75

Shen, S., Jiang, H., y Zhang, T. (2012). Stock Market Forecasting Using Machine Learning Algorithms. *Sruthi. V is currently pursuing BE computer Science and Engineering in SSN College of Engineering Chennai, India. She is doing research in the field of machine learning.*

Velásquez, J. D., Olaya, Y., & Franco, C. J. (2010). Predicción de series temporales usando máquinas de vectores soporte. *Ingeniare, 18*(1), 65-75.