UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
FACULTAD DE MATEMÁTICA, FÍSICA Y COMPUTACIÓN

TÉCNICAS DE APRENDIZAJE AUTOMATIZADO PARA EL PRONÓSTICO DE TEMPERATURAS MÍNIMAS EN EL CENTRO METEOROLÓGICO DE VILLA CLARA, SANTA CLARA

Trabajo de Diploma para optar por el

Título de Licenciado en Ciencia de la Computación

Autor

Swaabow Tsaone Thapelo

tthapelo@uclv.edu.cu

dabigswaabow@yahoo.co.uk

Santa Clara, 2014

UNIVERSIDAD CENTRAL “MARTA ABREU” DE LAS VILLAS
FACULTAD DE MATEMÁTICA, FÍSICA Y COMPUTACIÓN

TÉCNICAS DE APRENDIZAJE AUTOMATIZADO PARA EL PRONÓSTICO DE TEMPERATURAS MÍNIMAS EN EL CENTRO METEOROLÓGICO DE VILLA CLARA, SANTA CLARA

Trabajo de Diploma para optar por el

Título de Licenciado en Ciencia de la Computación

Autor

Swaabow Tsaone Thapelo

Tutoras

Dra. María del Carmen Chávez Cárdenas

Lic. Adis Perla Mariño Rivero

Consultante

Dr. Aldo S. Moya

Santa Clara, 2014

Hago constar que el presente trabajo de diploma fue realizado en la Universidad Central «Marta Abreu» de Las Villas como parte de la culminación de estudios de la especialidad de Ingeniería en Informática, y autorizo a que el mismo sea utilizado por la Institución para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

Firma del Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

Firma del Autor

Firma del Jefe del Laboratorio

To my Grand Parents

I dedicate this work to my late Grandparents Semaswe Kekgetetswe and Kekgetetswe Tumisi for their long lasting love and care they gave to me which made me to be whom I am at this moment. It's sad that they never lived long to see me become the man they always wished me to be.

May their souls rest in peace!

AGRADECIMIENTOS

Esta es la parte más difícil que me toca siempre a pesar de que disímiles jugaron un papel vital en mi vida y que con ayuda de ellos superé obstáculos y momentos difíciles en la vida desde que era niño hasta ahora como estudiante para lograr el principal objetivo, que es el de finalizar los estudios superiores.

Agradezco en primer lugar a Dios por haberme permitido lograr esta parte tan importante en mi vida. Sin el todo esto nunca sería posible.

A mi familia, a mis padres por darme la vida, por ser mi fortaleza en la vida y apoyarme siempre en todo desde que era niño sobre todo a mi abuelo K. Tumisi y mi abuela por ser mis ejemplos a seguir, por cuidarme, por todo el amor incondicional que me dieron, desvelo y dedicación que pusieron en hacerme un hombre, por sacrificarse por mí una y mil veces, por sufrir por mi bien estar, por hacerme feliz, por siempre estar ahí a mi lado cuando las necesitaba, por ayudarme a lograr mis sueños y creer en mí en todos los momentos.

A mis tías Cicilia, Oboletse, Kenny. A mis hermanos Velve, Spy, por creer en mí y apoyarme y por impulsarme en todos los momentos.

Agradezco a todos mis profesores que me han formado y de los cuales recibí la guía para desarrollar el Trabajo de Diploma con honradez y mucha atención, que me han aportado los más disímiles valores durante todos mis años de estudios.

A mi tutora, la Dra. María del Carmen Chávez Cárdenas por ser imprescindible en mi tiempo de estudios en la facultad, por apoyarme en todos los momentos, por sus palabras reconfortantes que me llegaban justo cuando las necesitaba, por ser el hombro en el cual me apoyaba al caerme, porque cuando tengo dificultades me anima con palabras que me hacen ver que siempre hay camino, por ayudarme a lograr mis sueños, por creer en mí, por ser la tutora que siempre se preocupaba por mí, por ser una excelente profesora y sobre todo por ser mi mentora.

A mi tutora Adis Perla Mariño Rivero por su valiosa ayuda, por su preocupación durante la culminación de la tesis y por ser una amiga que me ayudó en el desarrollo de este trabajo.

A la Dra. María Matilde García Lorenzo por su maravillosa sonrisa cuando me saluda y por ser mi profesora durante mis estudios.
A la profesora Mabel González Castellanos por sus consejos cuando yo estaba en primer año, y por ser mi guía en ese momento.
Al profesor Guillermo por toda su ayuda en mis estudios.
A Ricardo Valdés, por estar a mi lado, reír conmigo y compartir junto lo poquito que teníamos.
Al gran amigo mío el Monte, por ser mi amigo, mi tío, mi hermano, por preocuparse por mí y apoyarme en todos los momentos con lo que necesitaba durante mis estudios, para hacerme reír, para compartir conmigo durante los cinco años de mi carrera.
Agradezco a todas las personas, amigos que han hecho posible de que en estos momentos este por donde estoy, feliz por haber terminado mis estudios a pesar de todos los desafíos de la vida.

A mis amigos sobre todos Alejandro García Simón, por estar a mi lado siempre cuando le necesitaba, porque siempre tiene tiempo para mí. A Yuney por ser tan paciente conmigo, reír conmigo y apoyarme en todos. A Bac y Humberto por ayudarme siempre en lo que necesitaba.

A todos mis compañeros de estudio por ayudarme siempre que los necesité, por ser mis compañeros de estudio en esta carrera tan difícil, por divertirnos y reírnos juntos fuera de las clases, fiestas, festivales, y por ser mis amigos.

A mi amigo Milder de Angola, por compartir el tiempo conmigo y por divertimos juntos. A Osvaldo Manuel de Sousa, el "Gordo" de Angola, por divertirnos juntos y hacerme reír siempre, a Abdul, Trevor, Leatile Marata, Danny Bwalya, White, Snaxz, Marinice, Eva, Joanna, A todos aquellos que de una manera u otra me brindaron su apoyo incondicional.
A mis amistades Cubanos, Namibia, Vietnam, Granada, Angola, Brasil, Zambia, Sud-África y otros.

RESUMEN

Los pronósticos se mantienen como una de las tareas más desafiantes y fascinantes actualmente. Esto ha atraído la atención de los especialistas e investigadores de distintos campos, especialmente las ciencias computacionales. Estudios de la literatura muestran que las técnicas de aprendizaje automatizado mantienen un mejor rendimiento que los métodos convencionales. En este trabajo se proponen cuatro técnicas de aprendizaje automatizado: máquinas de vectores soporte, las redes neuronales artificiales, los procesos gaussianos y el aprendizaje basado en instancias. Estos modelos se han aplicado para predecir las temperaturas mínimas en cinco estaciones meteorológicas localizadas en la provincia de Villa Clara. Se desarrolló una aplicación SAGI V 1.0 en Java que emplea las técnicas propuestas en este trabajo.

La capacidad predictiva de estos modelos es evaluada en términos de: error medio absoluto y la desviación estándar con el objetivo de escoger el mejor modelo basado en reducción de errores durante la predicción de las temperaturas para una determinada estación meteorológica. Se realizaron diferentes configuraciones para cada modelo y se escogen las mejores configuraciones según su capacidad predictiva. Los resultados finales fueron comparados con aquellos obtenidos por el Sistema empleado actualmente en el Centro Meteorológico de Villa Clara, el MOS (*Model Output Statistic*s).

De esta comparación se destaca que las técnicas propuestas en este trabajo se pueden utilizar para conseguir una mejor predicción para este tipo de datos en los que existe una relación compleja. En cuanto al porciento de los casos positivos obtenidos por las nuevas técnicas, la capacidad predictiva del modelo SAGI resultó mejor con un 90.3% comparado con el 76.1% alcanzado por el modelo actual para el periodo Marzo 2011.

Palabras Clave: Temperaturas mínimas, el modelo estadístico MOS, Máquinas de vectores Soporte, las Redes Neuronales Artificiales, IBk, Procesos Gaussianos.

ABSTRACT

Prediction remains one of the fascinating yet challenging tasks at the moment. This has drawn a lot of interest especially for researchers from different fields and a lot has been done for the past years. Literature studies have shown that machine learning techniques achieved better performance than conventional methods. In this work four Machine Learning techniques: Support Vector Machine, Artificial Neuronal Networks, Gaussian Processes and Instance Based Learning (IBk) models are applied to make minimum temperature predictions for five weather stations located in the Villa Clara province. An application using these four techniques was developed using the Java language. The new application will be referee to as SAGI V 1.0 for simplicity.

Their predictive capability is evaluated in terms of Mean Absolute Error and Standard Deviation with the objective of choosing the better model based on error reductions during temperature predictions for a given weather station. Different configurations for each model were realized and the best ones were selected with regards to their predictive capability. The final results were compared to those obtained by the current system being deployed in the Villa Clara Meteorological Centre (CMPVC).

It was concluded that the proposed machine learning techniques can be deployed in the Villa Clara province with the aim of improving weather forecasting for this kind of data with complex relations. As to the positive case results obtained SAGI obtained 90% of success as compared to 76% obtained by the current system (MOS) being used in the CMPVC. The models were evaluated for the March 2011 winter season due to the fact that the relationship of the environmental variables is very complex during this period.

Key words: Minimum temperatures, statistical method MOS, Support Vector Machine, Artificial Neuronal Networks, IBk, Gaussian Processes.

Tabla de contenido

INTRODUCCIÓN

El pronóstico del tiempo constituye una de las cuestiones más relevantes y se ha convertido en uno de los problemas más desafiantes debido a su valor práctico en la Meteorología, comunicación, aviación, las series de consumo eléctrico, producción y agricultura, detección de sequía, la protección de vidas humanas y de recursos de vital importancia para la economía de los países, entre otros. Por tanto la modelación de variables y su posterior utilización en la predicción son una herramienta importantísima.
La temperatura constituye una de las variables ambientales más importantes para los seres vivos y el que mayor influencia ejerce en casi todos los factores de la vida del hombre, de los animales y de las plantas sobre la superficie terrestre. De las temperaturas depende en gran medida el estado de confort en que nos desenvolvemos día a día, a la vez que tienen una influencia marcada en el comportamiento de la humedad relativa. Estas dos variables en su conjunto son determinantes en el desarrollo de disímiles cultivos, a la vez que definen la aparición o no de numerosas plagas y enfermedades que frecuentemente atacan a las plantas [1-3]. Es importante la influencia de la temperatura en la salud humana, principalmente sobre aquellos pacientes que padecen enfermedades respiratorias [4]. Elevadas temperaturas afectan el confort del ganado vacuno, a la vez que es letal para el confort avícola. Por ello es necesario el máximo cuidado en las observaciones que han de reunir unas conclusiones indispensables.
Los factores fundamentales que modifican las temperaturas son la *humedad*, la *nubosidad*, las *precipitaciones* y la *fuerza del viento*. Las bajas temperaturas en Cuba están asociadas a la presencia de masas de aire con muy bajo contenido de humedad. El factor nubosidad juega un papel importante, pues la presencia de nubes bajas y medias en la bóveda celeste inhibe el enfriamiento nocturno (irradiación nocturna), produciendo un efecto "invernadero" sobre la zona afectada por la nubosidad. El descenso de las temperaturas en el horario de la madrugada depende en gran medida de la fuerza del viento. Las bajas temperaturas se asocian a velocidades muy bajas del viento o "calmas".
Todos estos elementos alimentan los modelos de pronóstico de temperaturas y se conocen como las variables del tiempo o variables meteorológicas.

ANTECEDENTES

Desde el año 2008 funciona en Cuba un modelo de pronóstico de temperaturas mínimas desarrollado por investigadores del Centro Meteorológico de Villa Clara, y fue modificado de nuevo en el año 2010. Este modelo de temperaturas está basado en el principio de predicción mediante el modelo MOS (*Model Output Statistics* en sus siglas en inglés) [5] y es empleado en el servicio operativo del Centro Nacional de Pronósticos y de los Grupos Provinciales como uno de los fundamentos para realizar los pronósticos de temperaturas para todo el país. El MOS es una técnica objetiva de pronóstico de tiempo que consiste en determinar una relación entre las variables exógenas y las variables endógenas mediante un modelo numérico. El método se basa en la incorporación de las salidas de los modelos hidrodinámicos de pronóstico tanto en la fase de desarrollo de las ecuaciones de regresión como en la aplicación operativa de las relaciones obtenidas [6]. Actualmente el modelo MOS emplea múltiple regresiones lineales como un método estadístico [7] aunque se podrían utilizar otros métodos estadísticos, por ejemplo: regresión polinomial o regresión logística y las redes neuronales.

En este caso se desarrollaron ecuaciones de pronóstico para cada estación de la red del Sistema Meteorológico Nacional y sobre esta base se construyeron los campos de las temperaturas mínimas para todo el archipiélago.

La expresión 1 muestra el prototipo de la ecuación de pronóstico de temperaturas mínimas y el parámetro "R" alcanzado a partir del proceso de Regresión para el período poco lluvioso, obtenidas para la estación meteorológica 78314, de la provincia de Pinar del Río.

$$Tn = -5.518 + 0.401 * N_{GFS} + 1.087 * Tn_{GFS} + 0.17 * U_{GFS} - 0.041 * V_{GFS} \quad (1)$$

$$R = 0.892 \quad (2)$$

donde

Tn: es la Temperatura mínima a pronosticar.

N_{GFS}: es la Nubosidad pronosticada por el GFS, correspondientes al día para el que se elabora el pronóstico.

Tn_{GFS}: es la Temperatura mínima para los períodos 09 – 15, correspondiente al día para el que se elabora el pronóstico.

U_{GFS}: es el Componente zonal del viento pronosticada por el GFS, correspondiente al día para el que se elabora el pronóstico.

V_{GFS} : es el Componente meridional del viento pronosticada por el GFS correspondientes al día para el que se elabora el pronóstico.

Trabajar con las componentes zonal y meridional del viento permite tomar en consideración no solo la fuerza del viento, sino su dirección.

Desventajas del MOS

Aunque el MOS es matemáticamente simple y poderoso tiene algunas limitaciones:

- El MOS no corrige un pronóstico malo.
- El MOS tiene muchas limitaciones importantes como las limitaciones en precisión para pronósticos a largo plazo y plazos extendidos.
- A veces los efectos de pronósticos para los terrenos locales son un problema para el MOS.
- El MOS puede resultar incorrecto si las condiciones son altamente irregulares.

Sin embargo, las relaciones estadísticas entre las salidas de los modelos hidrodinámicos y las variables medidas en la red de estaciones en Cuba, se buscó de manera lineal, con ayuda del método definido anteriormente. Precisamente, conociendo que la relación entre las variables que influyen en el comportamiento de las temperaturas mínimas, es complejo y no es lineal, los modelos hidrodinámicos sufren cambios continuamente que influyen en la calidad de los pronósticos realizados con las ecuaciones de regresión construidas previamente dando esto como una desventaja del modelo existente [6]. Esto implica que aunque los modelos estadísticos y los modelos numéricos de pronóstico existentes han sustituido los engorrosos e inexactos cálculos manuales de antaño, todavía la precisión de sus resultados está mediada. Continuar perfeccionando los resultados obtenidos mediante los modelos estadísticos y los modelos numéricos de pronóstico es un reto para la comunidad científica actual.

El campo de Aprendizaje Automatizado (AA) ha abierto un camino a las investigaciones de los sistemas complejos por la importancia de la precisión en el aspecto predictivo. Las técnicas de AA sirven entonces como alternativos fundamentos para problemas de predicción. Pues si bien el pronóstico de temperaturas mínimas es una tarea difícil, se piensa que un enfoque basado en aprendizaje automatizado, podrá

inferir un modelo útil y aplicable en el centro meteorológico de la provincia de Villa Clara, de manera tal que complemente los modelos existentes y aumentar el nivel de acierto en los pronósticos de las temperatura mínimas.

Dentro de las técnicas de aprendizaje automatizado que permiten hacer análisis de regresión se encuentran las Redes Neuronales Artificiales (RNAs), Máquina de Vectores Soporte (SVM), IBK (*Instance based learning* en sus siglas en inglés) y los Procesos Gaussianos (PG).

El concepto de las RNAs se originó cuando se quiere desarrollar un modelo matemático capaz de reconocer patrones complejos de la misma forma que las neuronas biológicas [8-9]. Las RNAs han mostrado ser eficientes en investigaciones relacionadas con la predicción con variables de tipo ambiental, contaminantes ambientales, entre otros. Las RNAs no requieren de conocimiento a priori del sistema bajo consideración y son muy apropiadas en la modelación de sistemas dinámicos en tiempo real. Por tanto, es posible montar un sistema de tal forma que estas puedan adaptarse a los eventos observados y por esto, son útiles en el análisis real.

Otra herramienta, que tiene también su fundamento en el AA son las SVM. Las SVM originalmente se emplearon para resolver problemas de clasificación o de reconocimiento de patrones. Una nueva versión de SVM para regresión fue propuesta en 1996 por Vladimir Vapnik, Harris Drucker, Chris Burges, Linda Kaufman y Alex Smola. En un inicio se le dio mayor difusión y aplicación a las RNAs en la solución de problemas con datos reales, de ahí que se considere a las SVM como un método novedoso. Las SVM se han aplicado al análisis de varios tipos de datos. Se ha empleado las SVM en el análisis de series de tiempo [10], en aplicaciones de pronóstico de tiempo [11] entre otros. De acuerdo a la literatura, la naturaleza de los datos es determinante en la estimación de los parámetros libres, dejando esto a quien implementa estas herramientas.

Los Procesos Gaussianos (*Gaussian Processes* o GP en sus siglas en inglés) son modelos probabilísticos no paramétricos usados principalmente para realizar aprendizaje supervisado como regresión y clasificación. Estos modelos también proporcionan un alto grado de confianza en sus estimaciones y predicciones. En los últimos años, este tipo de modelado probabilístico ha ganado gran aceptación en la comunidad de aprendizaje de máquina debido a la flexibilidad en su estructura

matemática unida a la facilidad en su implementación. Se han empleado estos modelos en diversos problemas [12].

IBk conocida como K-NN (K-*Nearest Neighbourhood*) es otra técnica de aprendizaje automatizado empleada en problemas de clasificación y regresión. La esencia del aprendizaje basado en instancias es retornar como solución a un problema, la solución conocida a un problema similar. El k-NN constituye un algoritmo clásico de esta forma de solución a problemas y ha sido empleado en problemas de clasificación y regresión.

El método básicamente consiste en comparar la nueva instancia a predecir con los datos o casos existentes del problema en cuestión, recuperando los k casos más cercanos, lo cual depende del parecido entre los atributos del nuevo caso con los casos de la muestra de aprendizaje o entrenamiento[13-14]. Como resultado del mismo se devuelve la clase mayoritaria de aquellos k casos más cercanos a él.

Planteamiento del Problema

Diversos métodos estadísticos se han desarrollado para dar solución al problema del pronóstico de las temperaturas mínimas. En el Centro Meteorológico de la Provincia de Villa Clara (CMPVC) se modelan las temperaturas mínimas mediante el MOS. La búsqueda de manera lineal de las relaciones estadísticas entre las salidas de los modelos hidrodinámicos y las variables medidas en la red de estaciones del país, con ayuda del modelo MOS dificulta las tareas de pronóstico meteorológico a corto y mediano plazo en el CMPVC, ya que los fenómenos meteorológicos son bien conocidos por tener comportamientos de tipo no lineal. El modelo convencional existente tiende a carecer de precisión para el pronóstico de temperaturas mínimas, al no poder considerar las características de las relaciones complejas entre variables.

Por otra parte, los investigadores en diferentes partes del mundo han empleado las técnicas de AA para resolver problemas del pronóstico de temperaturas. Sin embargo en el CMPVC, en el área de pronóstico de temperaturas, no se han aplicado estas aplicaciones operativas que ayuden a los servicios de pronósticos meteorológicos en la consecución de métodos más precisos y con mayor cobertura que los empleados actualmente.

Derivado de lo anterior, resulta de importancia el empleo de técnicas novedosas como las de AA para la predicción de las temperaturas mínimas en el CMPVC, ya que han

demostrado tener una buena capacidad predictiva en problemas de clasificación y regresión.

Para dar solución al problema de investigación se planteó el siguiente objetivo general de investigación que consiste en: Desarrollar una aplicación para el pronóstico de temperaturas mínimas en el CMPVC, mediante técnicas de Aprendizaje Automatizado, basado en las salidas de los campos de variables meteorológicas del Sistema Global de Predicción (GFS).

Objetivos Específicos:

1. Realizar un diseño experimental que permita obtener una propuesta de modelo para pronosticar las temperaturas mínimas en las distintas estaciones de la Provincia utilizando las técnicas SVM, K-NN, RNAs o Procesos Gaussianos.

2. Comparar la capacidad predictiva de los modelos de pronósticos propuestos para las estaciones de Villa Clara, con el modelo físico estadístico empleado actualmente en el CMPVC (MOS) y el GFS para el pronóstico de temperaturas mínimas.

3. Desarrollar una aplicación de software usando el enfoque que mejor resultado obtenga para una estación dada con el fin de aumentar el nivel de acierto en los pronósticos de la temperatura mínimas en el CMPVC.

Las preguntas de investigación planteadas son:

- ¿Cuál modelo usar de modo que garantice un balance adecuado entre eficiencia y eficacia en el pronóstico de corto y mediano plazos de temperaturas mínimas para las estaciones de la provincia de Villa Clara?
- ¿Los modelos obtenidos utilizando las técnicas de AA propuestas garantizan la predicción de las temperaturas mínimas en el CMPVC con resultados similares a los obtenidos por MOS y GFS?

Se formula las siguientes hipótesis de investigación:

Hipótesis

1. El desarrollo de pronósticos mediante el uso de sistemas que no estén sujetos a supuestos matemáticos (como por ejemplo supuestos de distribución) puede incrementar el grado de adaptabilidad que presenten estos, frente a los diferentes cambios observados en la atmósfera.

2. Los resultados de los sistemas computacionales empleados en los pronósticos meteorológicos podrían hacerse más precisos mediante la utilización de distintas técnicas de Aprendizaje Automatizado.

Las tareas de investigación trazadas son:

1. Estudiar los cuatros modelos, RNAs, SVM, IBK y Procesos Gaussianos incorporados en la plataforma WEKA para aproximación de funciones.

2. Realizar un estudio de los parámetros de los modelos RNAs, SVM, IBK y GP para obtener modelos de pronósticos de temperaturas mínimas en diferentes estaciones de la provincia de Villa Clara.

3. Diseñar e implementar una aplicación que interactúe con WEKA para el pronóstico de las temperaturas mínimas de las estaciones incluidas en el trabajo.

Métodos a emplear en la investigación.
Métodos empíricos: Para la presentación y discusión de los resultados;
Método experimental: Para comprobar la utilidad de los resultados obtenidos y la comparación con otros métodos reportados.

La novedad de la investigación radica en:
La nueva propuesta de pronosticar las temperaturas mínimas en el Centro Meteorológico de Villa Clara mediante técnicas de aprendizaje automatizado.

JUSTIFICACIÓN

Las técnicas de aprendizaje automatizado han sido aplicadas satisfactoriamente para resolver problemas tanto de clasificación como de regresión; no obstante, en el CMPVC no se han empleado estas técnicas para resolver problemas de pronóstico de tiempo. La aportación al campo de la Meteorología consiste en introducir estas técnicas como un marco potencialmente competitivo para pronosticar las temperaturas en el CMPVC. Así, se ha considerado trascedente emplear el enfoque del AA para resolver el problema de pronóstico de temperaturas mínimas en el CMPVC.

El estudio planteado ayudará, entre otros aspectos, a dotar a los Meteorólogos en CMPVC de otra alternativa para el pronóstico de temperaturas mínimas. No se intenta reemplazar con este enfoque computacional lo que con otros medios está probado con

calidad, sino el de aprovechar las características del mismo para fortalecer todo el proceso de pronóstico de temperaturas mínimas. Se trata de complementar a los modelos existentes para lograr obtener buenos resultados.

Estructura de la investigación

La tesis está estructurada en tres capítulos:

El Capítulo 1 se dedica a la elaboración del marco teórico desde el punto de vista del estudio del AA, específicamente se describen los algoritmos para dar solución al problema de regresión y se profundiza en los algoritmos RNAs, K-NN, SVM y Procesos Gaussianos.

En el Capítulo 2 se presenta una descripción y la organización de los datos del problema y los modelos de investigación. Luego se discute en detalle cómo se resuelve el problema de pronóstico mediante dichos modelos. Es decir, se abordan las herramientas a utilizar, diseño y la construcción de los modelos del sistema para el pronóstico de temperaturas mínimas en el CMPVC.

En el Capítulo 3, se presenta el análisis y validación del funcionamiento del sistema, los resultados y efectividad de los modelos propuestos para el pronóstico de temperaturas mínimas. Este documento culmina con las conclusiones y recomendaciones, se detallan las referencias bibliográficas y se incluyen algunos anexos para mostrar detalles complementarios.

Capítulo1: Marco Teórico

En este capítulo se presentan algunos conceptos importantes en que se fundamenta la teoría del aprendizaje automatizado y que están relacionados con los cuatro modelos de esta investigación. A lo largo de este capítulo se exponen los fundamentos básicos de las RNAs, SVM, K-NN y Procesos Gaussianos.

1.1. Aprendizaje Automatizado

La capacidad de aprender se considera como una de los atributos distintivos del ser humano y ha sido una de las principales áreas de investigación de la Inteligencia Artificial desde sus inicios. En los últimos años se ha visto un crecimiento acelerado en la capacidad de generación y almacenamiento de información, debido a la creciente automatización de procesos y los avances en las capacidades de almacenamiento de información. Debido a esto, se han desarrollado una gran cantidad de herramientas y técnicas que tienen que ver con el análisis de información. En este aspecto, el desarrollo en el área de AA ha sido fundamental. El área de AA en general trata de construir programas que mejoren su desempeño automáticamente con la experiencia.

El AA es el área de la Inteligencia Artificial que se ocupa de desarrollar técnicas capaces de aprender, es decir, extraer de forma automatizada conocimiento subyacente en la información. Constituye, junto con la estadística, el corazón del análisis inteligente de los datos [15].

El principio seguido en el AA es que la máquina genera un modelo a partir de ejemplos y lo usa para resolver el problema. Uno de los modelos de aprendizaje más estudiado es el aprendizaje inductivo, que engloba todas aquellas técnicas que aplican inferencias inductivas sobre un conjunto de datos para adquirir conocimiento inherente a ellos.

De forma más concreta, se trata de crear programas capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. En muchas ocasiones el campo de actuación del mismo se solapa con el de la Estadística, ya que las dos disciplinas se basan en el análisis de datos. Sin embargo, el AA se centra más en el estudio de la complejidad computacional de los problemas. Muchos problemas son de

clase *NP-hard*, por lo que gran parte de la investigación realizada en esta rama se enfoca al diseño de soluciones factibles a esos problemas.
El AA tiene una amplia gama de aplicaciones, incluyendo motores de búsqueda, diagnósticos médicos, detección de fraude en el uso de tarjetas de crédito, análisis del mercado de valores, clasificación de secuencias de ADN, reconocimiento del habla y del lenguaje escrito, juegos y robótica.

Algunos sistemas de este tipo intentan eliminar toda necesidad de intuición o conocimiento experto de los procesos de análisis de datos, mientras otros tratan de establecer un marco de colaboración entre el experto y la computadora. De todas formas, la intuición humana no puede ser reemplazada en su totalidad, ya que el diseñador del sistema ha de especificar la forma de representación de los datos y los métodos de manipulación y caracterización de los mismos

Clasificación del aprendizaje según los datos

El aprendizaje es el proceso por el cual el sistema modifica sus pesos en respuesta a una información de entrada, es decir, es el proceso por el que se produce el ajuste de los parámetros libres del modelo a partir de un proceso de estimulación por el entorno que rodea ese sistema. El aprendizaje es de particular interés, pues para que la red resulte operativa es necesario entrenarla para que sea capaz de realizar un determinado tipo de procesamiento o computo aprendiéndolo a partir de un conjunto de patrones de ejemplos, lo que constituye el modo aprendizaje.

Aprendizaje supervisado

El aprendizaje supervisado es una técnica para deducir una función a partir de datos de entrenamiento. El objetivo del aprendizaje supervisado es el de crear una función capaz de predecir el valor correspondiente a cualquier objeto de entrada válida después de haber visto una serie de ejemplos, los datos de entrenamiento. Los datos de entrenamiento consisten de pares de objetos (normalmente vectores): una componente del par son los datos de entrada y el otro, los resultados deseados.
Para este tipo se distinguen perfectamente rasgos predictores o rasgos de entrada y los rasgos objetivos o de salida, respondiendo estos a una etiqueta. La salida de la función

puede ser un valor numérico (como en los problemas de regresión) o una etiqueta de clase (como en los de clasificación).

Para ello, tiene que generalizar a partir de los datos presentados a las situaciones no vistas previamente. Este tipo de entrenamiento necesita un conjunto de datos de entrada previamente clasificado o cuya respuesta objetivo se conoce a modo de que el sistema tenga los objetivos como punto de referencia para evaluar su desempeño en base a la diferencia de estos valores y modificar los parámetros libres en base a esta diferencia. La tarea entonces es estimar una cierta función multivariable desconocida a partir de muestras tomadas aleatoriamente, por medio de la minimización iterativa de una función que representa el error esperado de la operación del sistema mediante alguna forma de aproximación.
De esta forma se dice que el aprendizaje es supervisado ya que el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar el sistema a partir de una entrada determinada. El supervisor controla la salida del sistema y en caso de que ésta no coincida con la deseada, se procederá a modificar los parámetros, con el fin de conseguir que la salida obtenida se aproxime a la deseada.

Aprendizaje no supervisado

Es un método de AA donde un modelo es ajustado a las observaciones. Los sistemas con aprendizaje no supervisado no requieren influencia externa para ajustar los parámetros, es decir no hay un conocimiento a priori. Todo el proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formado tan sólo por entradas al sistema. No se tiene información sobre las categorías de esos ejemplos, o sea la base de conocimiento del sistema está formada por ejemplos no etiquetados. Por lo tanto, en este caso, el sistema tiene que ser capaz de reconocer patrones para poder etiquetar las nuevas entradas. Una forma de aprendizaje no supervisado es la agrupación o *clustering*.

Aprendizaje semisupervisado

En este tipo se combinan los dos modos anteriores para poder clasificar de manera adecuada. Se tiene en cuenta los datos marcados y los no marcados.

Aprendizaje por refuerzo

Se trata de un aprendizaje supervisado que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, no se suministra explícitamente la salida deseada ante una determinada entrada. En este aprendizaje no se conoce la repuesta adecuada que debería presentar el sistema, pero dispone de algún mecanismo que indica si la repuesta es buena o no. El algoritmo aprende observando el mundo que le rodea. Su información de entrada es el *feedback* o retroalimentación que obtiene del mundo exterior como respuesta a sus acciones. Por lo tanto, el sistema aprende a base de ensayo-error.

1.1.1 Clasificación del aprendizaje según el tipo de conocimiento que se genera

El problema del aprendizaje se ha enfrentado desde diferentes perspectivas: el enfoque simbólico, el conexionista y el evolutivo. El enfoque simbólico puede ser inductivo o perezoso. El enfoque conexionista está orientado al aprendizaje cuando se resuelven problemas usando RNAs. El enfoque evolutivo ocurre en los algoritmos genéticos.

El AA, como hemos visto, permite resolver problemas mediante el empleo del conocimiento obtenido de problemas resueltos en el pasado, similares al actual. El problema del aprendizaje se ha enfrentado desde diferentes perspectivas: el enfoque simbólico, el conexionista y el evolutivo. La diferencia fundamental entre estas técnicas radica en la forma en que se almacena el conocimiento.

Así, en las RNAs, el conocimiento se traduce en una serie de pesos y umbrales que poseen las neuronas. En cambio, en el aprendizaje inductivo, el conocimiento se transforma en un árbol de decisión o un conjunto de reglas.

El aprendizaje inductivo

El aprendizaje inductivo se usa para adquirir conocimiento (formulado en forma de descripciones intencionales) a partir de ejemplos. El objetivo de la inducción es formular afirmaciones que explican los hechos dados y se pueden aplicar a hechos no vistos con anterioridad. Estas afirmaciones pueden expresarse como patrones, y estos patrones se representan por vectores, ecuaciones, árboles, reglas o enunciados lógicos. Muchos problemas de inducción se pueden describir como sigue. Se parte de un conjunto de entrenamiento de ejemplos preclasificados, donde cada ejemplo (también

llamado observación o caso) se describe por un vector de valores para rasgos o atributos, y el objetivo es formar una descripción que pueda ser usada para clasificar ejemplos previamente no vistos con alta precisión [13].

Aprendizaje perezoso

En el aprendizaje perezoso, al igual que en el aprendizaje inductivo, se aprende a partir de ejemplos, pero a diferencia de este:

i) Se usa descripción extensional, sin generar descripciones intencionales.

ii) El proceso de aprendizaje y el proceso de usar el conocimiento aprendido para resolver nuevos problemas no se separan.

iii) El paso de generalización se retrasa para la fase de solución de problemas.

Cuando se resuelve un nuevo problema P, la solución de un problema viejo se transfiere (quizás transformada) a P. Hay una generalización implícita entre el viejo y el nuevo problema. Un método clásico de aprendizaje perezoso es el algoritmo de los k-Vecinos más Cercanos

Métodos de aprendizaje perezoso bien conocidos son: el aprendizaje basado en instancias, el razonamiento analógico, y el razonamiento basado en casos.

Aprendizaje basado en instancias

La esencia del aprendizaje basado en instancias es retornar como solución a un problema, la solución conocida a un problema similar. Los algoritmos de aprendizaje basado en instancias se basan en ejemplos modelos; cada concepto se representa por un conjunto de ejemplos, cada ejemplo puede ser una abstracción del concepto o una instancia individual del concepto. Una extensión del aprendizaje basado en instancias consiste en usar los k casos más parecidos en lugar del más cercano, lo cual es apropiado cuando los ejemplos se describen mediante datos mezclados, no sólo mediante rasgos con dominio real.

Modelos conexionistas

Busca descripciones generales mediante el uso de la capacidad de adaptación de redes de neuronas artificiales. La red neuronal explora muchas hipótesis simultáneamente usando redes masivamente paralelas compuestas de muchos elementos de procesamiento conectados por enlaces con pesos. Los modelos de redes neuronales son

especificados por la topología de la red (estructura y tipo de enlaces), las características de los nodos (modelo de la neurona) y las reglas de aprendizaje (método de ajustar los pesos).

1.2 La predicción numérica o regresión

La predicción numérica o regresión consiste en dado un conjunto de variables predictoras o atributos predecir el valor numérico para una variable objetivo a partir de los valores predictores conocidos.

No existe un modelo de regresión mejor que otro de manera general; para cada problema nuevo es necesario determinar con cuál se pueden obtener mejores resultados, y es por esto que han surgido varias medidas para evaluar la calidad de los modelos de regresión y comparar los modelos empleados para un problema determinado [13]. Con el objetivo de validar la efectividad de un modelo de regresión y dar una idea de cuan efectivo es el mismo en la solución de un problema determinado al que se quiere aplicar, han surgido varias medidas para evaluar y comparar los modelos empleados para dicho problema.

1.2.1 Métodos para dar solución al problema de regresión

Diferentes métodos se han empleado para dar solución al problema de regresión entre los cuales aparecen métodos estadísticos como: regresión lineal, árboles de regresión, etc.; SVM, Procesos Gaussianos, métodos basados en modelos de RNAs como MLP, LVQ, etc. y métodos basados en instancias como k-NN. A lo largo del trabajo se denomina clasificador a cualquier modelo con capacidad de predecir un valor numérico (regresión). Por tanto se usaran los dos términos con el mismo significado.

Regresión lineal

En estadística la regresión lineal o ajuste lineal es un método matemático que modela la relación entre una variable dependiente Y, las variables independientes X_i y un término aleatorio ε.

Árboles de regresión

Es un árbol de decisión cuyas hojas predicen una cantidad numérica, ese valor numérico se calcula como la media del valor para la variable clase de todos los ejemplos que han llegado a esa hoja durante el proceso de construcción del árbol. La evaluación de un nuevo ejemplo es idéntico a los árboles de decisión, durante el proceso de predicción es posible utilizar un suavizado de los valores del ejemplo a tratar, con el fin de salvar las posibles discontinuidades presentes en los datos. El criterio de selección de una variable en la construcción del árbol está basado en una reducción del error esperado: reducción de la desviación o varianza en la variable objetivo. Al final el árbol se poda para evitar el sobreajuste.

Máquinas de vectores soporte

Las máquinas de vectores soporte (*Support Vector Machines*, SVMs) son un conjunto de algoritmos de aprendizaje supervisado desarrollados en los últimos años, partiendo de la teoría de aprendizaje estadístico y basada en el principio de minimización de riesgo estructural. Concretamente, fundamenta las decisiones de predicción o clasificación, no basadas en todo el conjunto de datos, sino en un número finito y reducido de casos, que constituyen los "vectores soporte". Se ha usado tanto para clasificación (aprendizaje supervisado con función objetivo discreta), como para regresión (aprendizaje supervisado con función objetivo continua) [16].

Dado un conjunto de ejemplos de entrenamiento (de muestras) podemos etiquetar las clases y entrenar una SVM para construir un modelo que prediga el rasgo objetivo de una nueva muestra. Intuitivamente, un SVM es un modelo que representa a los puntos de muestra en el espacio, separando las clases por un espacio lo más amplio posible. Cuando las nuevas muestras se ponen en correspondencia con dicho aproximador, en función de su proximidad puede ser predicho un valor objetivo.

La función Kernel

La manera más simple de realizar la separación es mediante una línea recta, un plano recto o un hiperplano N-dimensional.

Desafortunadamente los universos a estudiar no se suelen presentar en casos idílicos de dos dimensiones, sino que un algoritmo SVM debe tratar con

a) más de dos variables predictoras,

b) curvas no lineales de separación,

c) casos donde los conjuntos de datos no pueden ser completamente separados,

d) clasificaciones en más de dos categorías.

Debido a las limitaciones computacionales de las máquinas de aprendizaje lineal, éstas no pueden ser utilizadas en la mayoría de las aplicaciones del mundo real. La representación por medio de funciones kernel ofrece una solución a este problema. La función kernel se utiliza para hacer la proyección de datos que se encuentra en un espacio característico a un espacio representado en una dimensión superior el cual aumenta la capacidad computacional de las máquinas de aprendizaje lineal. Es decir, mapearemos el espacio de entradas X a un nuevo espacio de características de mayor dimensionalidad.

$$F = \{\varphi(x) \mid x \in X\} \tag{3}$$

$$x = \{x_1, x_2, x_3, \dots, x_n\} \rightarrow \varphi(x_1) = \{\varphi(x_1), \varphi(x_2), \varphi(x_3), \dots, \varphi(x_n)\} \tag{4}$$

Algunos de las funciones kernels más usados son:

1. Kernel Lineal

$$K(X, \acute{X}) = (< X, \acute{X} >) \tag{5}$$

2. Kernel Polynomial

$$K(X, \acute{X}) = (X, \acute{X})^{d} \tag{3}$$

3. Kernel de base radial

$$K(X, \acute{X}) = exp\frac{(||X - \acute{X}||)}{2\alpha^2} \tag{4}$$

K-NN.

La esencia del aprendizaje basado en instancias es retornar como solución a un problema, la solución conocida a un problema similar. La regla de clasificación o predicción por vecindad más general es la regla de los k vecinos más cercanos o simplemente k-NN. Se basa en la suposición de que los prototipos más cercanos tienen una probabilidad a posteriori similar. El método básicamente consiste en comparar la nueva instancia a predecir con los datos o casos existentes del problema en cuestión, recuperando los k casos más cercanos, lo cual depende del parecido entre los atributos del nuevo caso con los casos de la muestra de aprendizaje o entrenamiento. Como resultado del mismo se devuelve la clase mayoritaria de aquellos k casos más cercanos a él.

Este método consta de 3 pasos fundamentales. El proceso de aprendizaje de este aproximador consiste en almacenar en un vector el conjunto de entrenamiento, junto al rasgo objetivo asociado a cada muestra de este conjunto. Sea el conjunto de *n* casos o ejemplos E_i, cada E_i se describe por el vector de *m* rasgos predictores ($a_1(E_i)$, $a_2(E_i)$,..., $a_m(E_i)$) y un rasgo objetivo, donde $a_j(E_i)$ denota el valor del rasgo *j* en el ejemplo E_i. Sea *P*, definido como $(a_1(P),..., a_{m-1}(P))$ el objeto a predecir, entonces en primer lugar, y con motivo del aprendizaje del algoritmo, calcularemos la distancia euclidiana de cada muestra de entrenamiento, a todas las demás que están almacenadas en el vector del punto anterior y de las que conocemos la clase o valor objetivo a la que corresponden, quedémonos con las *K* muestras más cercanas y prediciendo la nueva muestra de entrenamiento.

$$d(P,E_i) = \sqrt{\sum_{k=0}^{n} w_i * (\partial(\, a_j(P\,), a_j(E_i)))^2} \tag{5}$$

$$\partial(x,y) = |x-y| \tag{6}$$

La segunda tarea para diseñar el modelo, es realizar el mismo proceso con los datos de validación Se calcula la medida sobre los ejemplos de este conjunto (desconocidos en la tarea de aprendizaje) para conocer su poder de generalización. Se conoce como mecanismo de aprendizaje perezoso (*lazy learning*). Debemos especificar una métrica

para poder medir la proximidad. Suele utilizarse por razones computacionales la distancia Euclidiana.

$$\begin{cases} \dfrac{\sum_{x \in K} y_x}{|K|} & \textit{si los casos k objetos cuenta igual} \\ \dfrac{\sum_x w_x y_x}{\sum_{v \in K} w_v} & \textit{si se tiene en cuenta un voto} \end{cases} \tag{7}$$

Dónde: *y* es el valor calculado para la función en el problema *P* e y_x es el valor objetivo en el ejemplo *x* que es uno de los *k* casos más cercanos (similares).

Medidas para determinar las instancias similares al problema.

En el paso uno de ambos algoritmos se pueden utilizar tanto distancias como funciones de semejanza. Obviamente, si se calculan distancias se seleccionaran los *K* ejemplos de menor distancia al problema, mientras que si se usan funciones de semejanza se seleccionaran los *K* ejemplos más similares.

1. Distancia Euclidiana

$$E(x,y) = \sqrt{\sum_{a=1}^{m} (x_a - y_a)^2} \tag{8}$$

2. Distancia de Chebychev

$$Ch(x,y) = \max |x_a - y_a| \tag{9}$$

3. Distancia de Manhattan

$$M(x,y) = \sum_{a=1}^{m} |x_a - y_a| \tag{10}$$

Ventajas del método k-NN:

Entre las ventajas del método se tiene que es de rápido aprendizaje y su capacidad de solución a partir de pocos ejemplos.

Limitantes del método k-NN:

El método k-NN tiene limitantes, por ejemplo: costo computacional que se incrementa con las dimensiones del conjunto de entrenamiento dado por el incremento del tiempo de respuesta, siendo la complejidad temporal O (*nd*), tal que O (*d*) es la complejidad de la distancia usada y O (*n*) lo que se requiere para explorar todo el conjunto de ejemplos y la complejidad espacial estará dada por la necesidad de almacenar los *n* ejemplos descritos por *m* rasgos.

Procesos Gaussianos

Los procesos Gaussianos son una generalización de distribuciones de densidad multivariadas Gaussianas a conjuntos de funciones infinitas continuas y se han utilizado para tareas de inferencia de datos al menos durante los últimos cien años. Los procesos Gaussianos mantienen estrecha conexión con las RNAs siempre y cuando ambos se estudien desde una perspectiva Bayesiana. Sin embargo, en contraste con las RNAs, los procesos Gaussianos tienen la ventaja de modelar de forma flexible y sin la limitación de tener que adaptar un gran número de parámetros, algo que comúnmente restringió la aplicación de RNAs en muchos problemas[12].

La clave a la hora de realizar con éxito un PG, es la selección del kernel adecuado. Los PG nos proporcionan una aproximación probabilística para problemas de regresión con

kernels. El modelo PG establece una relación entre los datos de entrada y la variable de salida, de la siguiente forma:

$$y = f(x) = \sum_{i=1}^{N} \alpha_i \mathrm{K}(\mathrm{X}_i, X) + \alpha_0 \tag{11}$$

Es posible manejar directamente la incertidumbre con respecto a los valores de la función en los puntos que nos interesan, lo que representa la perspectiva del espacio funcional (o PG) del problema. El punto clave de por qué se dejó de usar el enfoque basado en parámetros para modelar datos es que las proyecciones también se pueden manejar como variables aleatorias.

Las Redes Neuronales Artificiales

Bonifacio y Molina describen las RNAs como sistemas, hardware o software, de procesamiento, que copian esquemáticamente la estructura neuronal del cerebro para tratar de reproducir sus capacidades [17]. Las RNAs son herramientas matemáticas para la modelación de problemas, que permiten obtener las relaciones funcionales subyacentes entre los datos involucrados en problemas de clasificación, reconocimiento de patrones, regresiones, etc. Son consideradas excelentes aproximadores de funciones esencialmente no lineales, siendo capaces de aprender las características relevantes de un conjunto de datos, para luego reproducirlas en entornos ruidosos o incompletos.

Usualmente las RNAs reciben la información proveniente del exterior mediante un conjunto de neuronas de entrada y cuentan con un conjunto distinto, llamado neuronas de salida, para ofrecer los resultados. El resto de las neuronas se organizan en capas ocultas. Se concibe el cálculo general de la red a partir de la información que es procesada por cada una de sus neuronas en forma independiente. Cada una de ellas puede recibir información de las restantes y calcular su propia salida a partir de dicha entrada y de su estado actual, transitando eventualmente hacia un nuevo estado.

Por lo general, el flujo de cálculo de la red avanza progresivamente desde las neuronas de entrada hacia las neuronas de salida, en un proceso en el que cada una de las neuronas ocultas va activándose progresivamente según el esquema de conexión

particular de cada red [17-19]. El aprendizaje de los pesos se adapta en función del error cometido, y por tanto es suficiente con medir de forma adecuada el error.
Una RNAs se caracteriza por el modelo de la neurona, el esquema de conexión que presentan sus neuronas, o sea su topología, y el algoritmo de aprendizaje empleado para adaptar su función de cómputo a las necesidades del problema particular. Se ha producido una amplia variedad y clasificación de topologías de RNAs, que pueden agruparse en dos grandes grupos: las redes multicapa de alimentación hacia delante (*Feed-Forward Neuronal Networks*, FFN) y las redes neuronales recurrentes (*Recurrent Neuronal Networks*, RNN) [17-19].

Los algoritmos de entrenamiento constituyen métodos que se aplican sobre los modelos de red para ajustar sus pesos y obtener un comportamiento determinado. Con frecuencia los algoritmos de entrenamiento son caracterizados por la clase de topologías sobre las que se aplica, los tipos de parámetros libres que afecta (pesos de las conexiones entre neuronas, parámetros del algoritmo de entrenamiento, la topología misma de la red, etc.) y la regla de modificación de los mismos. Existe una amplia variedad de algoritmos de entrenamiento disponibles, y generalmente se clasifican en supervisados o no supervisados. El algoritmo de propagación del error hacia atrás (*Backpropagation*, BP), es el de más amplio uso, aplicado a redes con conexiones hacia adelante. Este algoritmo aplica la técnica gradiente descendente para la minimización del error de funcionamiento de la red [17, 20].

El *perceptron multicapa* (MLP) es una RNAs formada por múltiples capas, la cual tiene al menos una capa oculta con suficientes unidades no lineales que le permite aproximar cualquier tipo de función o relación continua entre un grupo de variables de entrada y salida, lo cual es la principal limitación del *perceptron simple*. Esta propiedad convierte a las redes MLP en herramientas de propósito general, flexibles y no lineales [17-18, 20]. Las redes MLP han sido aplicadas satisfactoriamente para solucionar alguna dificultad y diversos problemas entrenándolos de una manera supervisada con el algoritmo conocido como algoritmo del *error de la propagación hacia atrás*. Este algoritmo se basa en la *regla de aprendizaje por corrección de error*. De esta manera, la red tiene la capacidad de generalización: facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento.

Se han desarrollado RNAs que son utilizadas en meteorología para resolver problemas de predicción y clasificación [21-32], representando una técnica de modelización matemática que intenta imitar el proceso de aprendizaje que ocurre en el sistema nervioso.

Los tres conceptos clave de los sistemas nerviosos, que se pretende emular en los sistemas inteligentes, son: paralelismo, memoria distribuida y adaptabilidad, al entorno. De esta manera, podemos hablar de las RNAs como sistemas paralelos, distribuidos y adaptativos.

1.3 Conclusiones parciales del capítulo

El Aprendizaje Automático tiene una amplia gama de aplicaciones, entre las que se encuentran los problemas de la regresión. En regresión o la predicción numérica se tienen variables predictoras y un rasgo objetivo que en este caso es continuo.

En este capítulo se ha mostrado de manera breve una introducción al AA, los tipos de aprendizaje y se ha caracterizado cuándo se resuelve un problema de regresión para lo que se describen cuatro métodos de AA: las máquinas de vectores soporte, las redes neuronales artificiales, el método de los K vecinos más cercanos y los procesos gaussianos. Estos cuatro métodos se emplearán en el estudio experimental que se desarrolla en el segundo capítulo para proponer un modelo predictivo para distintas estaciones del CMPVC

Capítulo 2: DESARROLLO DE LA APLICACIÓN *SAGI V1*: Construcción de los modelos predictivos para el pronóstico de las temperaturas mínimas

Es de interés de este trabajo realizar el pronóstico de las temperaturas mínimas en el CMPVC mediante técnicas de aprendizaje automatizado. En el presente capítulo se modela el problema del pronóstico de las temperaturas mínimas en el CMPVC. Primero se presentan los materiales y métodos a emplear para lograr los objetivos y se describen las clases fundamentales para el diseño e implementación del software *SAGI V1* para el pronóstico de las temperaturas mínimas. Para esto se propone experimentar con cuatro técnicas de AA que permiten resolver problemas de regresión, ellas son: Máquinas de vectores soporte, las Redes Neuronales Artificiales, K-Vecinos más cercanos y los Procesos Gaussianos del aporte a la Meteorología está en introducir estas técnicas como un marco potencialmente competitivo para pronosticar las temperaturas en el CMPVC. Se utilizan datos provenientes del CMPVC para probar la calidad de las predicciones.

2.1 Introducción al problema del pronóstico de las temperaturas mínimas

Cuando se enfrenta un problema de clasificación o regresión se debe identificar y entender el problema, luego seleccionar el modelo clasificador, entrenarlo y por último validar o evaluar el mismo.

Para dar cumplimiento a los objetivos específicos planteados en esta investigación, se cuenta con un conjunto de bases de casos obtenidos del CMPVC. Se tiene como objetivo hacer una comparación entre los resultados que devuelvan los 4 modelos propuestos en esta investigación usando el software *Weka* (*Waikato Environment for Knowledge Analysis)*. En las bases de casos están presentes las variables que constituyen características o rasgos encontrados para determinar la temperatura mínima de la estación dada, con sus correspondientes instancias.

Problema: Dado un conjunto de variables ambientales predecir la temperatura mínima de una estación dada. Para este problema los meteorólogos consideran las características de las variables ambientales que se enumeran en la Tabla 2.1; las cuales constituyen un conjunto de datos históricos de las temperaturas mínimas comprendidas entre 2009 y 2011. Esta información fue obtenida del CMPVC. Todas ellas son consideradas como

atributos predictores del problema y poseen valores continuos. En total son 7 atributos predictores y un atributo objetivo.

Tabla 2.1 Atributos de la Base de casos para el pronóstico de las temperaturas mínimas

Atributo	Interpretación	Tipo
Día	el día para el que se elabora el pronóstico	Numérico
Mes	el día para el que se elabora el pronóstico	Numérico
Año	el día para el que se elabora el pronóstico	Numérico
Pron_Nub	Nubosidad pronosticada por el GFS	Numérico
U_Viento	Componente zonal del viento pronosticada por el GFS	Numérico
V_Viento	Componente meridional del viento pronosticada por el	Numérico
Pro_Tn	Temperaturas mínima pronosticada por el GFS	Numérico
Tn_Real	Temperaturas mínima real	Numérico

El día, Mes, Año correspondientes al día para el que se elabora el pronóstico y la Nubosidad pronosticada por el Sistema de Predicción Global (*Global Forecast System* o GFS en sus siglas en inglés), Componente zonal del viento pronosticada por el GFS y Componente meridional del viento pronosticada por el GFS, y la temperatura mínima pronosticada por el GFS también correspondientes al día para el que se elabora el pronóstico. La última variable corresponde a la variable objetivo o la temperatura mínima a pronosticar.

El dominio del rasgo objetivo se considera numérico. En este caso los valores que toman los rasgos son números de un universo finito. Por tanto se trata de un problema de regresión o la predicción numérica ya que todos los rasgos son de tipo numéricos [13].

El monitoreo de temperaturas mínimas se hace a lo largo del año pero se divide en dos etapas, verano e invierno.

2.2 Materiales y Métodos

2.2.1 Requisitos funcionales:

La verificación de los resultados concretamente se realizó aplicando la técnica de "comparación punto a punto" entre el pronóstico realizado y el valor real. En este caso, la verificación se realizó para cada una de las cinco estaciones meteorológicas de la provincia de Villa Clara con ayuda de las fórmulas 12-13; (error medio absoluto, AME) y la desviación estándar (STD). La desviación estándar puede ser interpretada como una medida de incertidumbre. La desviación estándar nos da la precisión de las medidas de los modelos: si la media de las medidas está demasiado alejada de la predicción (con la distancia medida en desviaciones estándar), entonces consideramos que las medidas contradicen la teoría.

$$AME_i = \frac{1}{N}\sum_{i=1}^{n} |[P_i - A_i]| \tag{12}$$

$$\mathrm{STD}_i = \sqrt{\frac{\sum_{i=1}^{n}(P_i - A_i)^2}{n-1}} \tag{13}$$

Donde:

AME_i y STD_i: Error medio absoluto y la desviación estándar respectivamente en los puntos a comparar.

P_i, A_i: Pronóstico y Temperatura Real respectivamente en cada nodo (i, j) de la malla.

N_i – Número de puntos a avaluar (coincide con la cantidad de estaciones meteorológicas de Cuba).

También se evaluó la efectividad de los modelos en "% de acierto", tomando como umbral de precisión el valor de $\pm$ 2, establecido en la actualidad en la "Metodología de evaluación de los pronósticos", del Sistema Nacional de Pronósticos del Instituto de Meteorología de Cuba[33]**.** Se evaluó el período Marzo de 2011 que corresponde con la etapa invernal, en que los pronósticos de las temperaturas mínima se hace más complejo. Finalmente se realizó una comparación de los resultados generales de los modelos que se proponen con los que se obtienen con el modelo MOS.

Figura 2.1 Metodología de verificaciones los resultados

JAVA

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. Java tiene una gran herencia del lenguaje C, aunque con una gran simplificación en los aspectos de bajo nivel ya que no es necesario el tratamiento de punteros, ni de memoria [34].

La plataforma Weka está desarrollada en el lenguaje de programación Java. Su gran versatilidad radica en que no se necesita ningún sistema operativo en concreto para que funcione, utiliza una máquina virtual que puede ser instalada en cualquier sistema operativo y dispositivo, incluyendo dispositivos móviles [34]. En el año 2006 se convierte en un lenguaje con licencia libre por lo que su utilización aumenta, así es como surgen aparte de las librerías (API) propias del código, otras que son de máxima utilidad, en nuestro caso hemos utilizado el API de Weka [14].

WEKA

Para el desarrollo de este trabajo se trabajará con el API de Weka empleando la plataforma de Weka en su versión 3.7.5, cuyo nombre proviene de *Waikato Environment for Knowledge Analysis*, es un paquete de software libre, desarrollado en la Universidad de Waikato, Nueva Zelanda. Está desarrollado en el lenguaje de programación Java y se distribuye bajo los términos de la licencia GNU (*GNU is Not*

Unix) [14, 35]. Ejecuta sobre cualquier plataforma y ha sido probado en los sistemas operativos Linux, Windows y Macintosh. Weka expone una extensa colección de algoritmos genéricos implementados en Java, útiles para ser aplicados mediante las interfaces que ofrece o para ser encapsulado dentro de cualquier aplicación a través de su API (*Application Programming Interface* en sus siglas en inglés).

Weka contiene herramientas para realizar transformaciones sobre los datos, tareas de clasificación, regresión, agrupamiento, asociación y visualización. El hecho de que provee una interfaz uniforme para acceder a los mismos, permite que sus usuarios puedan comparar los resultados de aplicar diferentes métodos a un problema en cuestión y seleccionar aquellos que sean más apropiados para dicho problema.

La API de Weka proporciona a los programadores un alto nivel de interoperabilidad, integración y reusabilidad de las funcionalidades de la plataforma, con el objetivo de extenderla, o para integrar funcionalidades ya implementadas y probadas en proyectos independientes, a través de código Java.

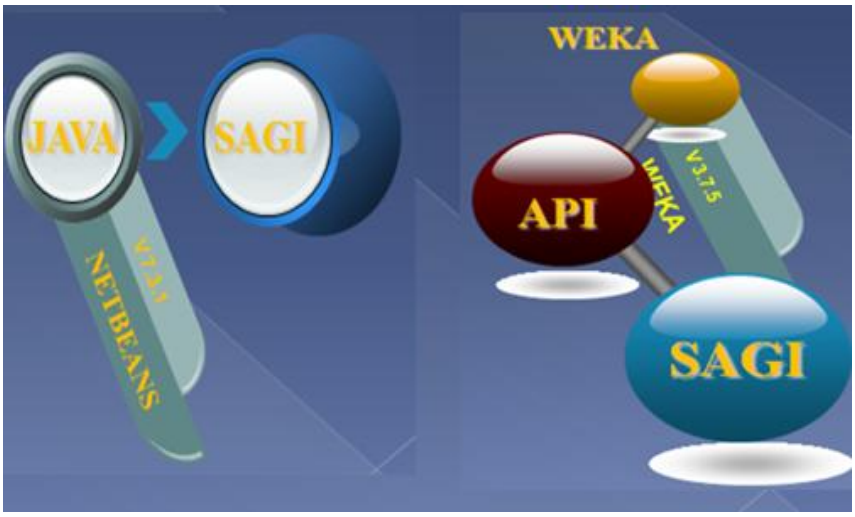

Figura 2.2 Materiales para el desarrollo de la aplicación SAGI

Weka está constituida por paquetes generales que se extienden de la raíz denominada Weka, y estos a su vez contienen subpaquetes. Los paquetes principales son los siguientes:

- ✓ core: Contiene las clases e interfaces que conforman el núcleo de Weka.
- ✓ gui: Paquete con la implementación de las interfaces gráficas.
- ✓ associations*:* contiene las clases que implementan los algoritmos de asociación.

✓ classifiers*:* agrupa todas las clases que implementan algoritmos de clasificación y éstas a su vez se organizan en subpaquetes de acuerdo al tipo de clasificador.

✓ experiment*:* agrupa las clases controladoras que permiten la realización de experimentos con varias bases de casos y diferentes algoritmos.

✓ attributeSelection*:* contiene las clases que implementan técnicas de selección de atributos.

✓ estimators*:* contiene las clases que realizan estimaciones (generalmente probabilísticas) sobre los datos.

De las funciones disponibles en la API de Weka para el subpaquete classifiers se utilizan en el trabajo las siguientes (ver tabla 2.2):

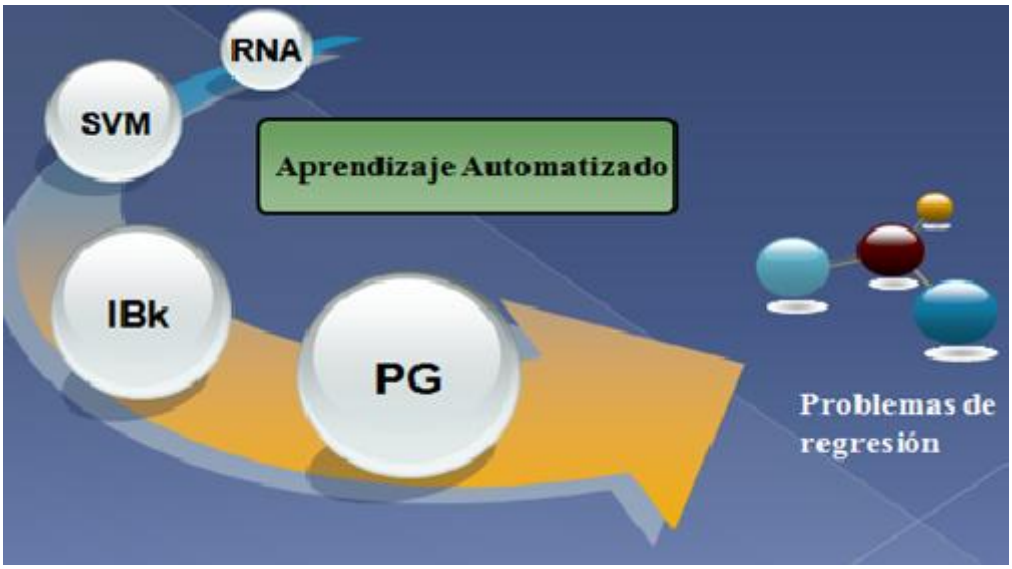

Figura 2.3 Los cuatros clasificadores propuestos en este trabajo.

Tabla 2.2 Nombres con los que parecen los clasificadores incorporados en Weka

Clasificador	**Nombre en Weka**	**Carpeta**
Máquinas de Vectores Soporte	SMO	functions
Multilayer Perceptron	MultilayerPerceptron	functions
k-vecinos más cercanos	IBk	lazy
Procesos Gaussianos	GaussianProcess	functions

Trabajo con estructuras de datos en Weka

El trabajo con estructuras de datos se realiza a través de las clases básicas: Instance, Attribute. La clase instance define una estructura de datos basada en filas y columnas, los atributos se corresponden a las columnas y las filas contienen un conjunto de instancias o ejemplos (core.Instance) que se almacenan conteniendo los valores de sus atributos.

Opciones para la generación y evaluación de un clasificador.

Todos los algoritmos de clasificación en Weka son ubicados en el paquete classifiers y heredan de la clase Classifier. Para entrenar el clasificador con un juego de datos brinda la opción buildClassifier (Instances data).

El paso de la evaluación, incluye la colección de parámetros estadísticos que se pueden configurar en la clase Evaluation, proporcionando el método evaluateModel (Classifier c, Instances data) para la evaluación con un juego de datos dedicado y el método crossValidateModel(Classifier c, Instance data, Integer numFolds, new Random(Interger seed)) para realizar la validación cruzada sin distribuir. Para proporcionar los resultados de la evaluación, existen implementados varios métodos como toSummaryString() y toClassDetailsString().

Los archivos de las bases de casos que utiliza Weka tienen extensión ".arff" y deben cumplir con la estructura mostrada en la Figura 1.3, de no cumplirse con ella, se notifica la incompatibilidad. En esta investigación se trabajará con el formato ARFF para los datos de entrada, lo que no significa que sea el único que admita Weka. Esta herramienta tiene intérpretes de otros formatos como CSV, que son archivos separados por comas o tabuladores (la primera línea contiene los atributos) y C4.5 que son archivos codificados según el formato C4.5, donde los datos se agrupan de tal manera que en un fichero *.names* estarán los nombres de los atributos y en un fichero *.data* los datos en sí.

Conjuntos de validación

La forma en que se dividen los datos en conjunto de entrenamiento y prueba es también muy importante. La mejor forma de organizar el experimento en entrenamiento y

prueba realmente depende de las características de la base de conocimiento. Muchos de los algoritmos disponibles en Weka antes de ser usados necesitan especificar qué datos serán usados como conjunto de entrenamiento y cuáles cómo conjunto de prueba [13-14]. Estos son conjuntos disjuntos, el primero de ellos utilizado por los algoritmos de AA para realizar el proceso de aprendizaje fijando sus parámetros (ejemplo, los pesos de una RNA), mientras que los ejemplos del conjunto de prueba se utilizan para medir el desempeño del algoritmo.

El método *boostrap* se basa en la generación de *n* conjuntos de cardinalidad *I* desde el conjunto de datos original, con reemplazo [13].
A continuación se exponen algunos variantes que se suele emplear para la resolución de problemas de clasificación y predicción numérica.

1. Usar los mismos datos como conjunto de entrenamiento y de prueba, conformado con todas las instancias del archivo de datos (*Use training set*). Esta técnica es la más vieja y simple que se basa en entrenar y probar el clasificador con la misma base de conocimiento. Este método puede traer como consecuencia un sobre-aprendizaje del clasificador, o sea, que el clasificador más que generalizar el conocimiento de los datos, aprenda estos "de memoria", pues no se tiene en cuenta cómo reacciona el modelo ante casos que no ha visto antes.

2. Realizar una validación cruzada estratificada con un número de particiones dado (*k-fold cross-validation*). Es uno de los más usados, este método se basa en dividir la base en k particiones y realizar k procesos de entrenamientos y pruebas, donde el proceso i se basa en tomar la partición i para prueba y el resto para entrenamiento. Se dice estratificada porque cada una de las partes conserva las propiedades de la muestra original (porcentaje de elementos de cada clase).

3. Tomar un porcentaje del conjunto de datos como muestra de entrenamiento y lo restante como muestra de prueba (*Percentage split*). Esta técnica es conocida como el método H (*Hold-out*) y también como *Percentage split*, que divide la base, en un porciento para entrenamiento y otro por ciento para prueba o muestra de control. Una versión de éste es el *Data Shuffle* que realiza n veces el método H y promedia los resultados. Este método no es efectivo para bases pequeñas, pues los ejemplos pudieran

no ser representativos si se diera el caso de que los casos no fueran divididos convenientemente [13-14, 35].

4. Usar todas las instancias del conjunto de datos solamente como conjunto de entrenamiento, y proporcionar como conjunto de prueba un nuevo archivo de datos (*Supplied test set*). Esta técnica se emplea cuando se cuenta con suficiente cantidad de datos para el entrenamiento y para el control o prueba [13, 35].

5. Actualmente se usa también en vez de dos conjuntos de datos, tres: uno para entrenamiento, uno para prueba y un tercero para validación. Este último se usa como seudoentrenamiento, de tal manera que el proceso de entrenamiento se detiene cuando comience a decrecer el rendimiento sobre el conjunto de validación, aunque continúe aumentando sobre el conjunto de entrenamiento. Este método es muy útil para evitar el sobreentrenamiento. También se usa para ajustar parámetros y seleccionar un modelo apropiado. Tiene como desventaja que necesita un conjunto de datos muy grande [13].

2.3 Diseño de la Aplicación SAGI V1.0

En esta sección se describe el proceso del diseño de la aplicación SAGI V1.0 (Support vector machine, Artificial neuronal networks, Gaussian processes and IBk). Este proceso se lleva a cabo mediante una serie de pasos que son; la construcción de los modelos, el entrenamiento y el control. Finalmente se muestra una breve introducción de la aplicación diseñada.

Weka es un sistema de código abierto, una de sus grandes ventajas es facilitar el desarrollo de sistemas reutilizando clases ya implementadas en esta herramienta. Cuando para un juego de datos se han probado los diferentes algoritmos y variantes de preprocesamiento de los datos disponibles, con la configuración seleccionada se puede desarrollar un software a la medida del usuario. Esto significa que reutilizando el código implementado se desarrolla un software, que independiza al usuario final de esta herramienta. En este epígrafe se ejemplifica lo anterior mediante un sistema para pronosticar las temperaturas mínimas de la estación dada, utilizando los nuevos modelos propuestos los cuales se guardan para hacer futuras predicciones.

La implementación de la aplicación consta de tres clases; *Trainner*, *Predictor* y SAGI_GUI. A continuación se describen las funcionalidades de las tres clases utilizadas.

Trainner: Clase representativa del objeto entrenador, donde un entrenador se caracteriza por un modelo y la base de casos a entrenar. Esta clase tiene como constructor el método "*entrenador*()" que permite entrenar el modelo dado. La clase también permite guardar un objeto del modelo para luego usarlo.

Predictor: Clase que tiene como método principal "*pronosticar*()". Esta clase permite cargar el modelo recién guardado y usarlo en la predicción. El método *pronosticar*()es el encargado de predecir el valor numérico de un nuevo caso a partir de un conjunto de atributos de una cierta instancia o un caso dado. En ella se hace uso del objeto de la clase *classifyInstance()* que permite predecir el valor continuo de una instancia.

SAGI_GUI: Clase que construye la interfaz gráfica de la aplicación y también facilita la captura de los datos de entradas del usuario. Es en esta clase donde se inicializan las dos clases descritas anteriormente.

El diagrama de clases fundamental para implementar *la aplicación SAGI se puede ver en el* Figura 2.4.

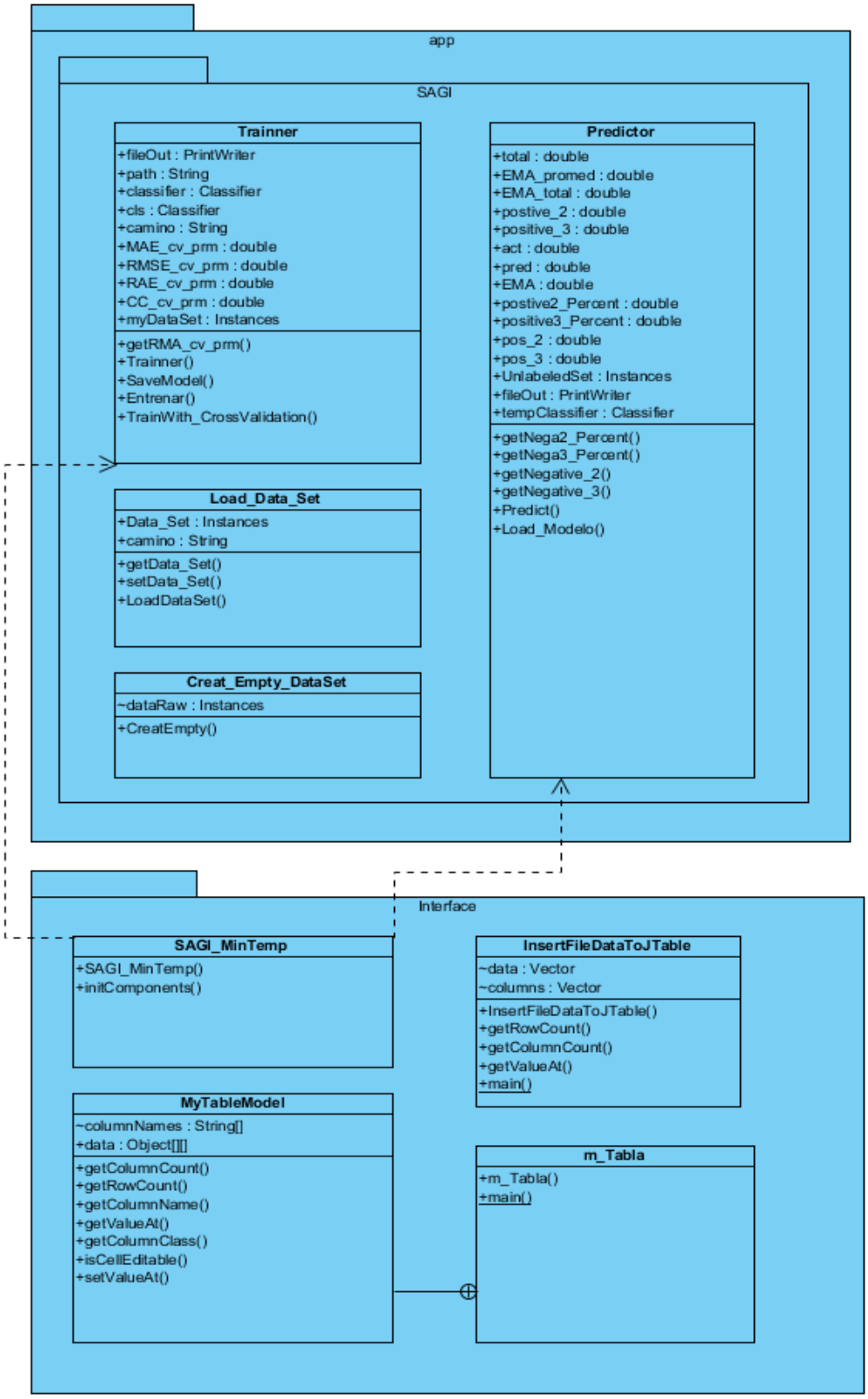

Figura 2.4 Diagrama de clases de la aplicación SAGI

2.3.1 Construcción y Entrenamiento de los modelos

En esta sección se describe el proceso de la construcción de los modelos, el entrenamiento, el control y finalmente se explican cómo se evaluaran los cuatros modelos propuestos en este trabajo.

Construcción de los Modelos

A la hora de diseñar un sistema para que emplee las funcionalidades del Weka en problemas de clasificación o regresión, es importante tener en cuenta la clase abstracta *Classifier*, que es la que se utiliza para dar todas estas facilidades. Esta clase, que es la más importante en el paquete *classifiers* constituye una superclase de todos los clasificadores existentes, tiene definido los métodos principales que debe tener un aproximador o clasificador. En regresión hay que redefinir 2 métodos que son los que a continuación describimos:

buildClassifier(): se encarga de la construcción del modelo del aproximador tomando como parámetro las instancias de entrenamiento. Debe inicializar todas las variables que correspondan a las opciones específicas del esquema. Nunca debe modificar ningún valor de las instancias. Después de terminada la ejecución de este método el clasificador debe ser capaz de predecir la clase de cualquier instancia nueva y en el caso del aproximador de predecir el valor continuo para cualquier instancia nueva.

classifyInstance(): permite clasificar o predecir el valor continuo de una instancia concreta. En el caso de la clasificación devuelve la clase en la que se ha clasificado o "desconocido" si no se consigue clasificar. Este algoritmo está diseñado de manera general también para regresión, de forma que en este caso devuelve un valor continuo.

Para el análisis se han comparado cuatro clasificadores o esquemas con diferentes configuraciones: RNA, KNN, SVM y GP. El software se apoya en las bibliotecas de clases implementadas en Weka, usando solamente las clases de los modelos propuesto en este trabajo y tomando las facilidades para la lectura de ficheros. Los constructores por defecto de los 4 clasificadores se muestran en la Figura 2.5. Estos clasificadores reciben opciones que les permiten obtener distintos resultados dado una base de conocimiento. Para usarlos basta importar GaussianProcesses, MultilayerPerceptron y SMOreg contenidos en el paquete functions de classfiers y IBk contenido en el paquete lazy. Por ejemplo: import weka.classifiers.functions.SMOreg;

```
MultilayerPerceptron MLP_classifier = new MultilayerPerceptron(); //MLP
IBk  KNN_classifier = new IBk();  //K-NN
SMOreg SVM_classifier = new SMOreg();  // Support Vector Machine para Regresiones
GaussianProcesses PG_classifier = new GaussianProcesses();  //Procesos Gaussianos
```

Figura 2.5 Los constructores de los 4 clasificadores

Una vez invocados los métodos esenciales para el funcionamiento del modelo, se necesita la implementación o invocación de otros métodos auxiliares para facilitar el trabajo con la interfaz del usuario. En particular, se debe invocar el método *setOptions()* para que sea posible establecer los parámetros del entrenamiento por parte del usuario (Ver Anexo 3), de lo contrario el clasificador usará las opciones que trae por defecto.

```
mlp_Options = "-L 0.4 -M 0.8 -N 500 -V 0 -S 0 -E 50 -H \"3, 0\" -D";
MLP_classifier.setOptions(weka.core.Utils.splitOptions(mlp_Options));
…
```

Figura 2.6 Cuadro que muestra el paso de los parámetros del modelo PG.

De esta forma es posible capturar los datos de entrada del usuario, pasarlos al método *setOptions()* y luego estos se usan como parámetros del entrenamiento del modelo en cuestión. En la Figura 2.6 se puede observar cómo se pasan las opciones al clasificador MLP_clasifier.

A la hora de diseñar un sistema para que emplee las funcionalidades del Weka para problemas de regresión es importante tener en cuenta la clase abstracta *Classifier*, que es la que se utiliza para dar todas estas facilidades. En regresión hay que redefinir 2 métodos que son los que a continuación describimos:

buildClassifier(): se encarga de la construcción del modelo tomando como parámetro las instancias de entrenamiento. Debe inicializar todas las variables que correspondan a las opciones específicas del esquema. Nunca debe modificar ningún valor de las instancias. Después de terminada la ejecución de este método el clasificador debe ser

capaz de predecir la clase de cualquier instancia nueva y en el caso del aproximador de predecir el valor continuo para cualquier instancia nueva.

Construcción del Modelo *Gaussian Processes*

El clasificador GaussianProcesses (GP) admite como parámetros el kernel y el parámetro de ruido (noise). La clave a la hora de realizar con éxito el GP, es la selección del kernel adecuado. Dentro de los kernels empleados por el aproximador GP del Weka 3.7.5 para problema de predicción numérica se encuentran; el Puk, PolyKernel, NormalizedPolyKernel, y el RBFKernel.

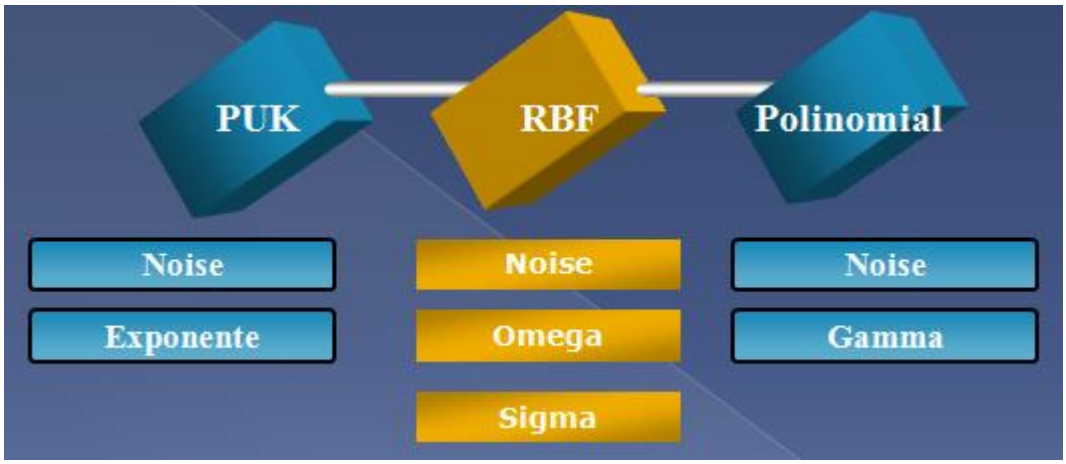

Figura 2.7 Los tres kernels propuestos para el modelo PG

El kernel Puk (Pearson VII Universal Kernel) tiene la posibilidad de cambiar fácilmente mediante la adaptación de sus parámetros. Esta única propiedad lo convierte en un kernel genérico que puede remplazar el conjunto de kernels previamente mencionados [36]. B. Ustun, W.J. Melssen y L.M.C. Buydens demostraron que ya no es necesario realizar una selección de las distintas funciones kernels, lo que simplifica el proceso de construcción de modelo, ahorrando el esfuerzo y el tiempo de desarrollo del modelo [36] .

Dentro de las opciones principales del kernel Puk se encuentran el tamaño de la cache (*cacheSize*), el valor de Omega (omega) y el valor de Sigma (sigma).

Estos datos se capturan por la interfaz y se pasan al método *setOptions*(), despuésse construye el clasificador del entrenamiento.

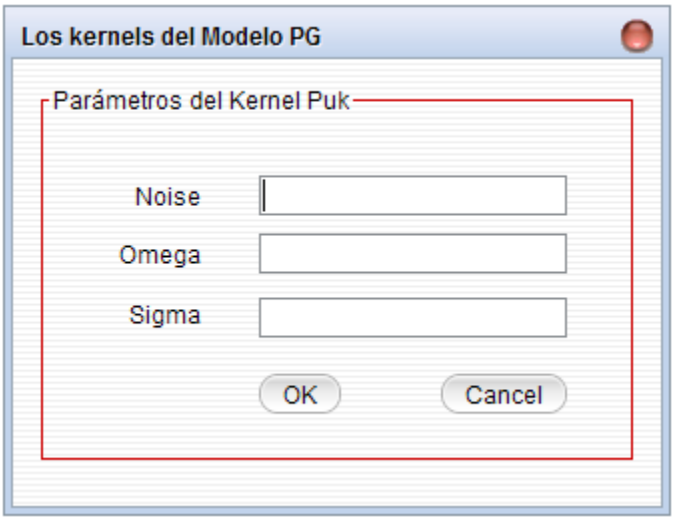

Figura 2.8 Subcuadro de diálogo *Parámetros de kernel Puk* del GaussianProcesses.

```
my_Noise= Noise; //Valor capturado
my_Omega = Omega; //Valor capturado
my_Sigma = Sigma; //Valor capturado
PUK_Kernel = " -N 0 -K weka.classifiers.functions.supportVector.Puk -C
250007";
PG_Options=
my_Noise.concat(PUK_Kernel).concat(my_Omega).concat(my_Sigma);
PG_classifier.setOptions(weka.core.Utils.splitOptions(PG_Options));
…
PG_classifier.buildClassifier(myDataSet);
…
```

Figura 2.9 Cuadro que muestra el paso de los parámetros del modelo

Se puede observar que solo se capturan los parámetros que tienen mucho efecto en el valor objetivo a la hora de cambiar su valor y los restantes se asumen que forman parte del kernel ya que la variación de estos, no tienen efecto en el valor objetivo. Esta suposición facilita el trabajo para el usuario de la aplicación.

Luego de pasar las opciones al clasificador, este seconstruye mediante el método *buildClassifier()* tomando como parámetro las instancias de entrenamiento '*myDataset*'. Después de terminada la ejecución de este método, el clasificador GP_classifier es capaz de predecir el valor continuo para cualquier instancia nueva.

Construcción del Modelo SVM

El clasificador SMOreg implementa una máquina de vectores soporte para regresión y admite como parámetros: el kernel, el parámetro de complejidad (C), y el algoritmo de aprendizaje (I). Se pueden emplear varios algoritmos para aprender los parámetros, los algoritmos se seleccionan poniendo el valor de *RegOptimizer* y el algoritmo más popular es el *RegSMOImproved*. Este clasificador emplea los mismos kernels usados por el clasificador GaussianProcesses del Weka 3.7.5 para problemas de predicción numérica.

De la misma forma se capturan los datos y se pasan las opciones al clasificador, luego se construye el modelo mediante el método *buildClassifier(),* como se explicó para el clasificador *GaussianProcess* previamente. El subcuadro de dialogo que captura los datos se muestra en la Figura 2.10.

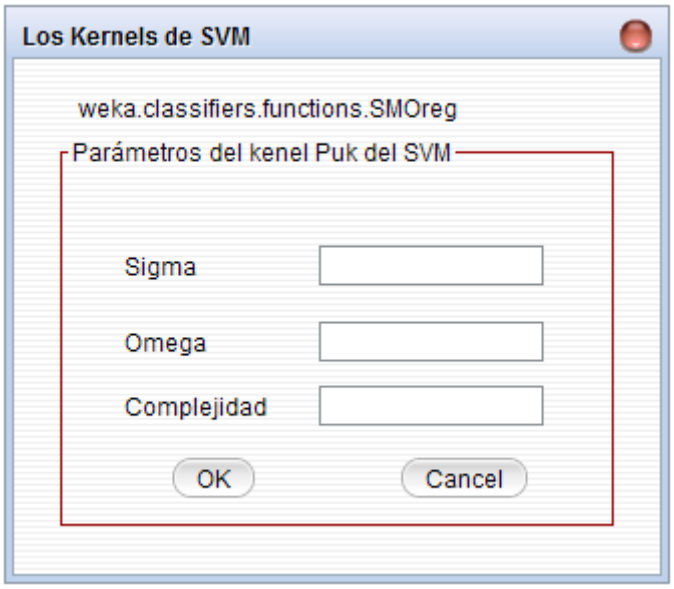

Figura 2.10 Subcuadro de diálogo *Parámetros del kernel Puk* del SVM

Construcción del Modelo IBk

Este clasificador se conoce como K-Vecino más cercano y se encuentra en el paquete lazy. Para importarlo basta incluir '*weka.classifiers.lazy.IBk*'. Admite como parámetros: la cantidad de vecinos a emplear (*K*), el método para los pesos de las distancias (*distanceWeighting*), y el algoritmo para búsqueda de los vecinos, el cual trae el *LinearNNSearch* por defecto. También se puede seleccionar apropiadamente el valor de

K empleando la medida de validación cruzada lo cual se especifica con –X en las opciones.

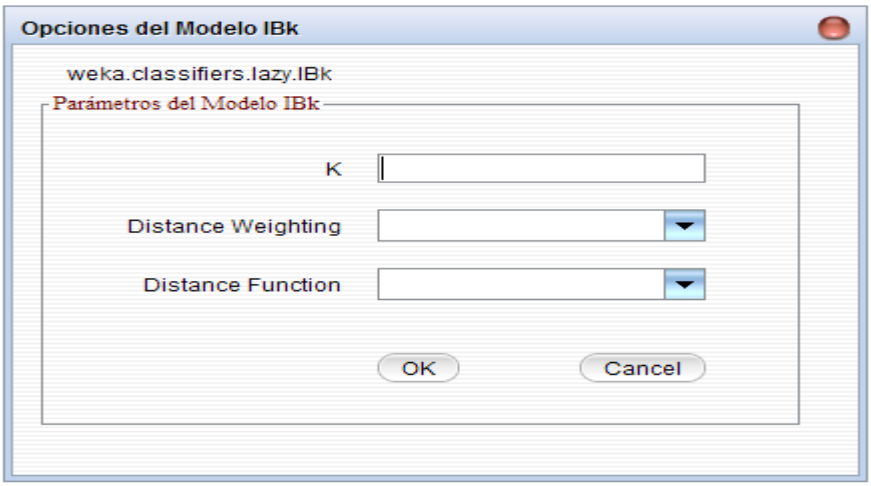

Figura 2.11 Subcuadro de diálogo *Parámetros del modelo IBK*

Construcción del Modelo MLP

Para usar este clasificador basta importar su paquete mediante *import weka.classifiers.functions.MultilayerPerceptron*. Este clasificador utiliza la propagación hacia atrás para clasificar las instancias.

learningRate – la razón por la cual se actualizan los pesos.

momentum – el valor aplicado a los pesos durante la actualización de estos.

trainingTime – el número de iteraciones para el entrenamiento.

hiddenLayers: define las capas ocultas de la red. La lista contiene los valores enteros positivos uno por cada capa oculta y se separan por comas. El valor 0 indica que no habrá una capa oculta.

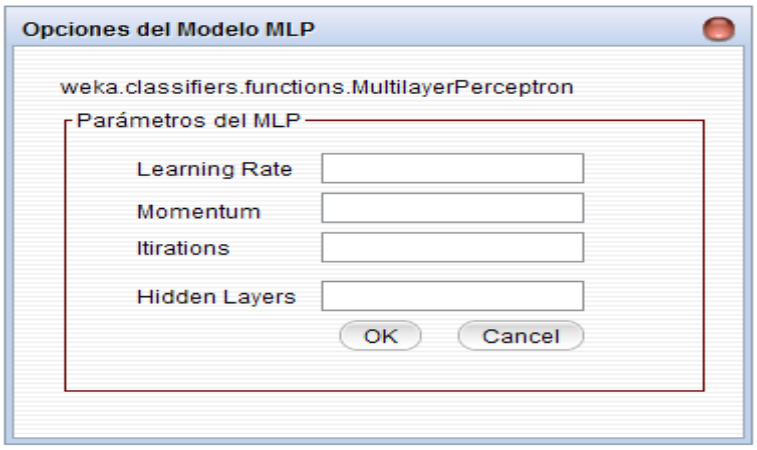

Figura 2.12 Subcuadro de diálogo *Opciones del Modelo RNA*

Entrenamiento de los clasificadores

El paso que sigue después de la construcción de los clasificadores es el de entrenamiento. Primeramente el clasificador es entrenado en el constructor de la clase, aquí se construye una instancia de la clase *Instances*, solamente se necesita pasar el fichero donde se encuentre el conjunto de datos. El constructor de *Instances* se encarga de leer los datos del fichero y almacenarlos en el objeto *Instances*. Para cargar los archivos en nuestro programa se emplean la clase *DataSource* que permite cargar archivos con distintos formatos o extensiones. También se puede emplear una clase específica si se conoce el formato del archivo de datos de entrada. A continuación se muestra como se emplea esta clase.

Seguidamente se utiliza este objeto para entrenar los cuatro algoritmos a utilizar, SVM, MLP, IBk y GP, llamando al método *buildClassifier* y pasándole el conjunto de entrenamiento ya construido como se explicó previamente.

```
import weka.core.converters.ConverterUtils.DataSource;
import weka.core.Instances;
...
Instances my_DataSet = DataSource.read("/address/dataset.arff");
Instances data2 = DataSource.read("/address /dataset.csv");
Instances data3 = DataSource.read("/address /dataset.xrff");
...
// Se define el ultimo atributo como la variable objetivo
if (data.classIndex() == -1)
data.setClassIndex(data.numAttributes() - 1);
...
```

Figura 2.13 Empleo de la clase *instances* para lectura de archivos.

Después del entrenamiento el clasificador está listo para clasificar la nueva instancia. Se toman los valores entrados mediante la interfaz visual y se construye un objeto de la clase *Instances* pasándole estos valores. El objeto construido se pasa al método *classifyInstances* de las clases *SVM*, *MLP*, *PG* e *IBk*, que retorna el valor de la clase en que se clasifica la instancia.

La ventaja que ofrece Weka para el desarrollo de este tipo de aplicaciones es fundamentalmente la rapidez con que pueden ser implementadas.

2.4 Diseño de los experimentos

La presente sección describe los experimentos que se realizaron para comparar el desempeño predictivo de los algoritmos propuestos.

En el primer experimento se utilizaron RNAs con algoritmo de aprendizaje "backpropagation" y todas las neuronas (capas ocultas) utilizan la función "sigmoide" excepto la neurona en la capa de salida que emplea la función de activación lineal [14]. La selección del número de neuronas y capas ocultas resulta una cuestión poco clara al momento del diseño de una RNAs. Demasiadas unidades pueden llevar a poca capacidad de generalización. Por otro lado, pocas unidades pueden llevar a que la RN no posea la capacidad suficiente para resolver el problema en cuestión.

Se probaron tres topologías de red para observar el comportamiento del MAE (*Error Medio Absoluto*) en función del número de neuronas y de capas ocultas. En la primera se utilizaron 7 neuronas de entrada correspondientes a los atributos predictores y una capa oculta con una sola neurona (7,1,1). En la segunda se utilizaron 7 neuronas de entrada y 3 neuronas en la capa oculta (7,3,1). En todos los casos, la capa de entrada estuvo formada por 7 neuronas correspondientes a los rasgos predictores mientras que la capa de salida estuvo formada por una neurona que representó la temperatura mínima a pronosticar. Se utilizaron distintas tasas de aprendizaje, coeficientes de momento, y cantidad de iteraciones de forma tal de obtener los mínimos valores en el MAE. La figura siguiente muestra las distintas configuraciones de las redes**.**

Tabla 2.5 Configuración del modelo MLP.

Modelo	L	M	N	H
MLP 1	0.08	0.04	500	7,1,1
MLP 2	0.04	0.1	500	7,3,1
MLP 3	0.04	0.1	500	7,4,1
MLP 4	0.08	0.04	1500	7,1,1
MLP 5	0.08	0.04	2500	7,3,1
MLP 6	0.04	0.1	2500	7,4,1

En el segundo experimento se emplearon los Procesos Gaussianos con tres kernels (núcleo).

Se utilizaron tres kernels para problemas de regresión:

- el kernel Puk con el parámetro del ruido (Noise), Omega y Sigma,
- el kernel RBF con dos parámetros (Noise y gamma).
- el kernel Polinomial también con dos parámetros (Noise y gamma).

Tabla 2.6 Configuración del kernel Puk del modelo GP.

Parámetros	Puk1	Puk2	Puk3	Puk4	Puk5	Puk6	Puk7	Puk8	Puk9	Puk10
Noise	0.4	0.1	0.4	1.0	0.4	0.6	0.4	0.4	0.2	0.6
Omega	13.0	14.5	14.5	15.5	1.0	0.4	0.5	0.6	0.6	2.5
Sigma	4.2	3.5	3.5	1.0	1.0	0.8	5.5	5.5	4.5	3.5

Tabla 2.7 Configuración del kernel RBF del modelo PG.

Parámetros	RBF 1	RBF 2	RBF 3	RBF 4	RBF 5	RBF 6	RBF 7	RBF 8	RBF 9	RBF 10
Noise	0.2	0.06	0.1	1.0	0.04	0.04	0.08	0.01	0.006	0.04
gamma	0.008	0.0008	0.0008	0.001	0.0008	0.0006	0.0006	0.0008	0.008	0.008

Tabla 2.8 Configuración del kernel polinomial del modelo PG

Parámetros	PolyK 1	PolyK 2	PolyK 3	PolyK 4	PolyK 5	PolyK 6	PolyK 7	PolyK 8	PolyK 9	PolyK 10
Noise	0.8	0.8	0.8	0.8	0.8	0.6	0.6	0.6	0.6	0.6
Exp	1	2	3	4	5	1	2	3	4	5

Para el modelo IBk se emplea la función de validación cruzada para encontrar el K óptimo para cada base de casos, y luego se emplean estos para el entrenamiento de los modelos. Se emplean las dos distancias pesadas que trae el Weka y como medidas para determinar las instancias similares al problema se emplean la distancia Euclideana y la distancia de Manhattan.

Tabla 2.9 Configuración del modelo IBk

Distancia	K =3	K = 4	K =5	K =6	K =8	K =9	K =18
1/dist	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana
1-dist	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana	Euclidiana
1/dist	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan
1-dist	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan	Manhattan

Por último para el modelo SVM nada más se emplean el kernel Puk con sus parámetros respectivos; el parámetro de ruido (Noise) y el parámetro omega. El kernel Puk es un kernel universal y genérico empleado actualmente para problemas de regresión y clasificación. Este kernel es capaz de reemplazar los otros kernel que hemos visto hasta ahora como se mencionó en el epígrafe 2.3.1. Para este usaremos el algoritmo *RegSMOImproved* como optimizador. Se varían también el parámetro de complejidad C de forma tal de obtener los mínimos valores en el error.

Tabla 2.10 Configuración del modelo SVM

Parámetros	Puk 11	Puk 12	Puk 13	Puk 14	Puk 15
C	1.0	1.5	2.0	1.2	1.6
Noise	0.001	0.0001	0.080	0.008	0.01
Omega	1.0	1.8	1.2	1.4	1.5
Sigma	1.0	1.5	0.4	0.6	0.6
Tolerancia	0.001	0.0001	0.002	0.004	0.006

2.5 Conclusiones parciales del capítulo

En este capítulo se plantea el problema del pronóstico de las temperaturas mínimas a resolver y se explican las variables a utilizar. Se selecciona los materiales a utilizar para la implementación de la aplicación, además se analizaron las clases que se implementaron con el fin de interactuar con la plataforma Weka. Mediante el API de Weka se logra diseñar una aplicación con dos funcionalidades principales cuales son: el entrenamiento de los modelos y el pronóstico de los valores objetivos de las temperaturas mínimas.

Se describe las configuraciones empleadas para los cuatro modelos escogidos para resolver el problema y se explica brevemente el diseño del software SAGI V 1.0.

Capítulo 3: ¿CÓMO UTILIZAR LA APLICACIÓN SAGI? ANÁLISIS DE RESULTADOS.

3.1 La Aplicación SAGI V1.0

La aplicación tiene una única ventana principal y varios subcuadro de diálogo. Cuenta con dos funcionalidades principales: para entrenar y para pronosticar. La funcionalidad de entrenar permite al usuario entrenar el modelo con una base de casos y guardarlo para luego realizar la acción de pronosticar. La funcionalidad de pronosticar le permite al usuario capturar los datos de entrada y hacer pronóstico del caso presentado. El sistema se implementa de tal forma que cargue automáticamente el modelo adecuado cuando se selecciona la estación de la cual se va a pronosticar el nuevo caso.
A continuación ofrecemos una breve guía de cómo utilizar la aplicación desarrollada para la experimentación.

Descripción

SAGI V1.0 *(Support vector machine, Artificial neuronal networks, Gaussian processes, Instance based learning*), es un ambiente de desarrollo integrado, que permite el entrenamiento de los clasificadores y la ejecución de bases de casos para hacer pronóstico de temperaturas mínimas. El ambiente permite editar, compilar y ejecutar bases de conocimiento.

Interfaz gráfica de usuario

El entorno gráfico utiliza diferentes menús y cuadros de diálogo que facilitan el trabajo de los usuarios finales.
La Figura 3.1 muestra la ventana principal del sistema, puede apreciarse (de izquierda hacia derecha):

- la barra de título con la identificación del sistema,
- la barra de opciones que brinda acceso a las funcionalidades del sistema,
- la barra de Entrenar el Modelo que brinda la funcionalidad de entrenar el modelo dado una base de casos de una estación.

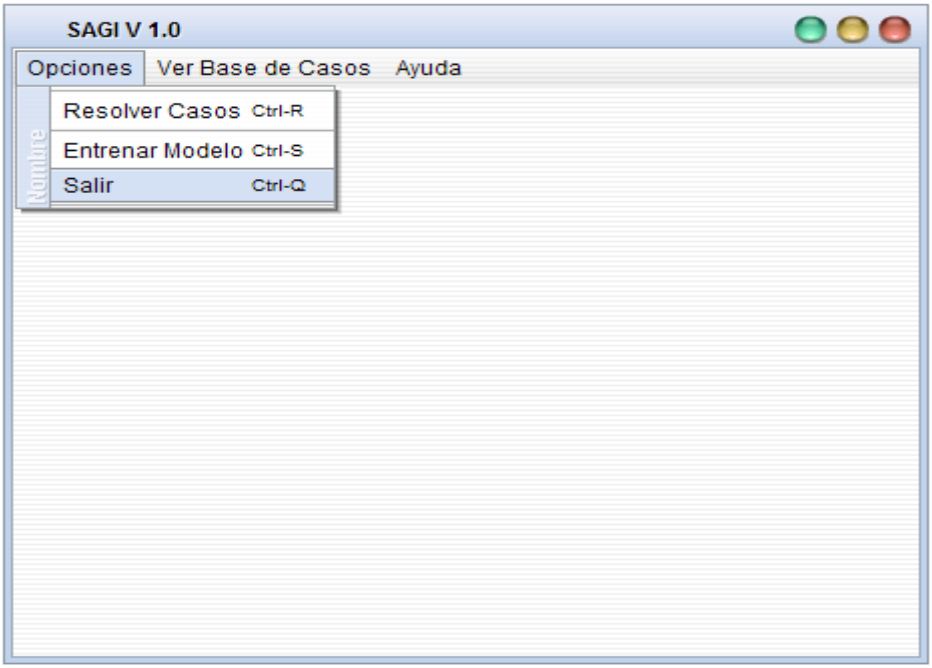

Figura 3.1 Vista inicial del Software SAGI V 1.0.

El primer paso, una vez que se ejecuta la aplicación SAGI V1.0 es seleccionar la opción *Entrenar Modelo* en el Menú Opciones y entonces se muestra un diálogo para seleccionar las opciones de los modelos a entrenar como se ve en la Figura 3.2. Luego de seleccionar el modelo se procede seleccionar los kernels y ajustar sus parámetros correspondientes para cada kernel. Para ello se hace eso de los Radio button que aparecen en la ventanas del modelo actual o mediante un solo botón si el modelo no tiene muchas opciones como es en el caso del modelo IBk y el modelo MLP (ver Figura 3.3).

Una vez terminado este proceso de configuración se carga la base de casos a usar, por lo cual se debe oprimir el botón **Cargar la base de casos** y luego se entrena el modelo mediante el botón **Entrenar el modelo actual**.

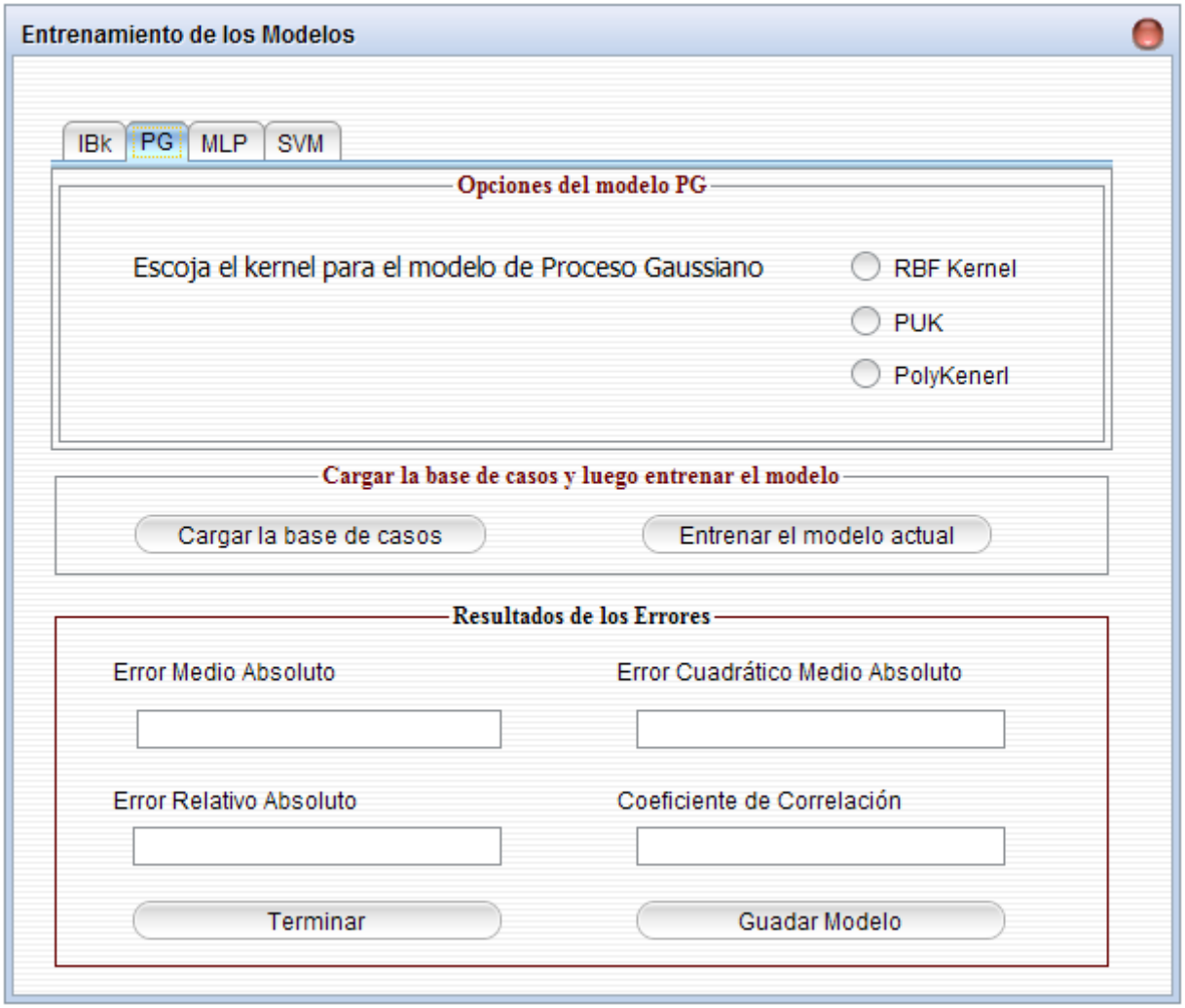

Figura 3.2 Ventana para la selección del modelo de entrenamiento

Una vez terminado el proceso de entrenamiento, el sistema ofrece algunas estadísticas en el cuadro **Resultados de los errores**. Se muestra el error medio absoluto, el coeficiente de correlación, el error cuadrático medio absoluto y el error relativo absoluto (ver Figura 3.4). El sistema además permite usar el botón **Guardar Modelo** para guardar el modelo para luego usarlo en la predicción.

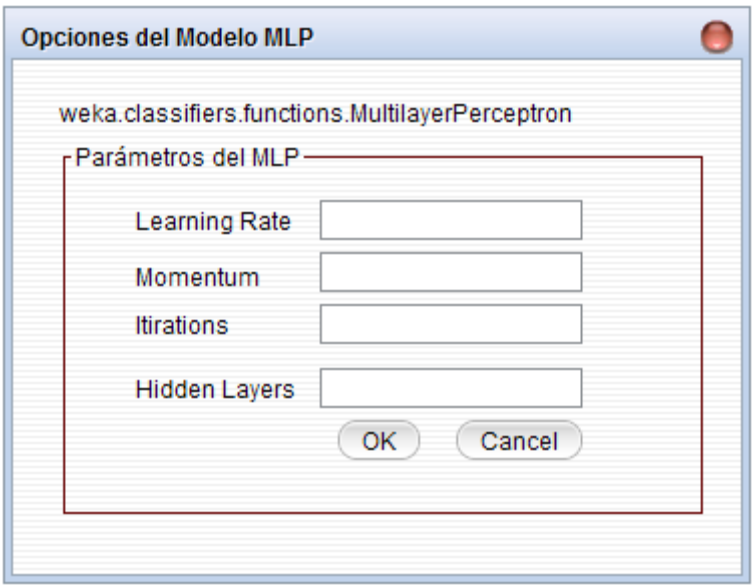

Figura 3.3 Ventana para la edición de los parámetros de entrenamiento del modelo MLP.

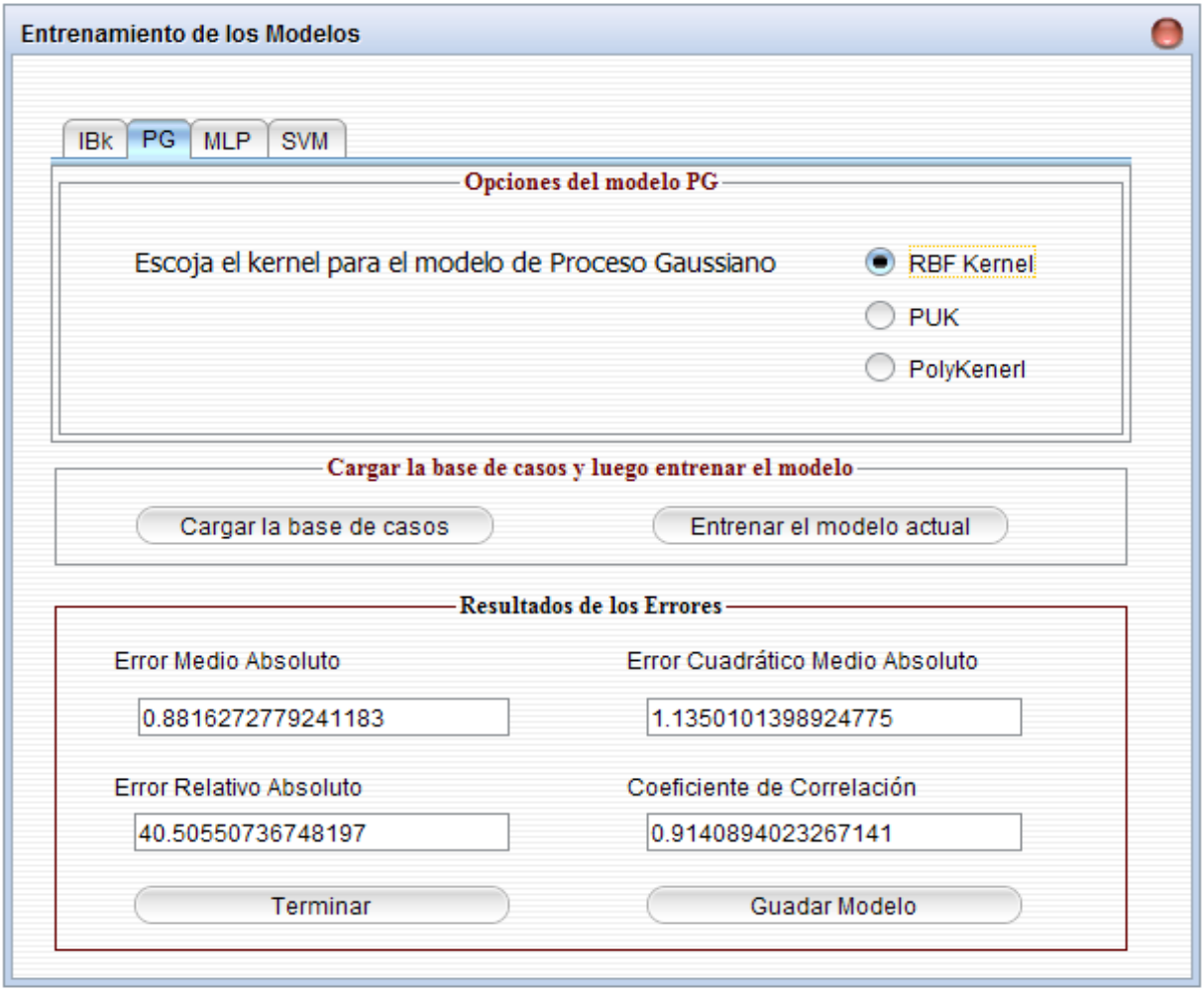

Figura 3.4 Ventana para los resultados de los errores del modelo de entrenamiento.

3.2.1 Predicción de un nuevo caso de una estación dada

La segunda funcionalidad de la aplicación es hacer pronóstico del valor objetivo o sea predecir la temperatura mínima de una estación dado los datos de entrada o predecir las temperaturas mínimas dado un conjunto de casos de entradas. A continuación se describen los pasos a seguir para este proceso.

El primer paso luego de seleccionar la opción ***Opciones*** en el Menú es seleccionar la opción **Resolver Casos**. Este abre un cuadro de dialogo que permite al usuario seleccionar el tipo de pronóstico que puede ser un solo caso (una instancia) o un conjunto de instancias como se mencionó previamente. Para este proceso se debe oprimir el combo box señalado con la etiqueta **Tipo de caso** como se ve en la Figura 3.5 y luego se carga el modelo usando el botón **Cargar el Modelo** para hacer el pronóstico.

Par continuar con el proceso se presiona el botón **Continuar** y se muestra un diálogo que nos permite suministrar a la aplicación los datos de entrada que son; el día, el mes, el año, el componente nubosidad (Nubosidad), el componente zonal de viento (U_Viento), el componente meridional del viento (V_Viento), y la temperatura mínima pronosticada por el GFS (Tn_Pron) que en total son 6 valores numéricos. La nubosidad y los componentes del viento también son valores pronosticados por el GFS correspondiente al día por el cual se realiza el pronóstico. Luego se realiza el pronóstico mediante el botón **Pronosticar** y los resultados se muestran en el cuadro señalado con el nombre **Resultados** y en él se muestra el valor pronosticado representado por la etiqueta ***Tn_Min*** como se ve en la Figura 3.6. Finalmente se pueden guardar los resultados para luego usarlos o simplemente cancelar el proceso completo.

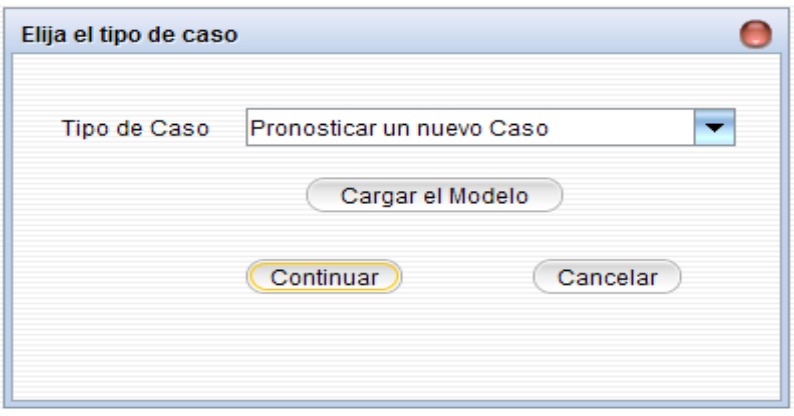

Figura 3.5 Ventana que muestra la selección del tipo caso a realizar.

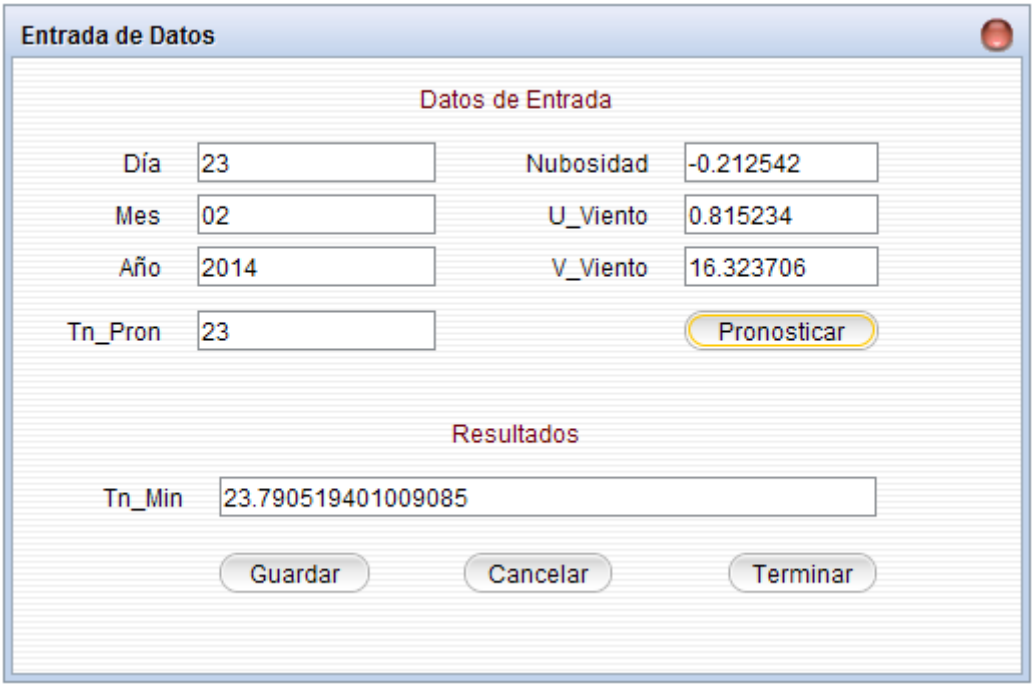

Figura 3.6 Ventana que muestra los resultados de la acción de pronosticar un nuevo caso.

3.3 Predicción de k- casos de una estación dada

Se procede similarmente al caso descrito anteriormente, lo que en lugar de seleccionar la opción Pronosticar un nuevo caso se selecciona la opción Pronosticar K-Casos y luego se carga el modelo correspondiente a la estación por el cual se hace el pronóstico como se ve en la Figura 3.7. Al oprimir el botón Continuar se muestra una ventana que nos permite cargar la base de casos o sea los K-Casos a predecir y esto se hace mediante el botón Cargar la base de casos (Ver Figura 3.8). Finalmente se oprime el botón Pronosticar y luego se muestran los resultados obtenido por el modelo en el cuadro señalado con la etiqueta Resultados de los K-Casos. Los resultados muestran el nombre de la estación, el número total de los casos (K) representado por la etiqueta Total: Casos, la cantidad de los casos con el error medio absoluto menor o igual que el umbral 2 (Total: Casos Positivos2), la cantidad de los casos con el error medio absoluto menor o igual que el umbral 3 (Total: Casos Positivos3), el porciento de acierto de estos casos; % Casos Positivos2 y Casos Positivos3 respectivamente.

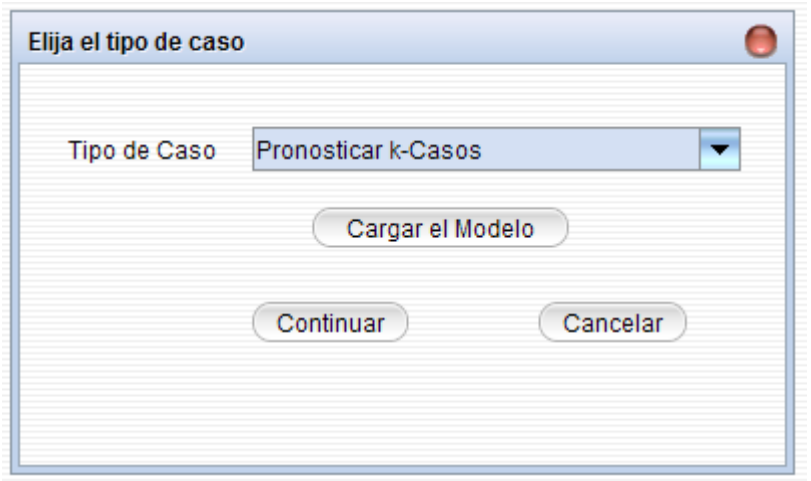

Figura 3.7 Ventana que muestra la selección del tipo caso a realizar.

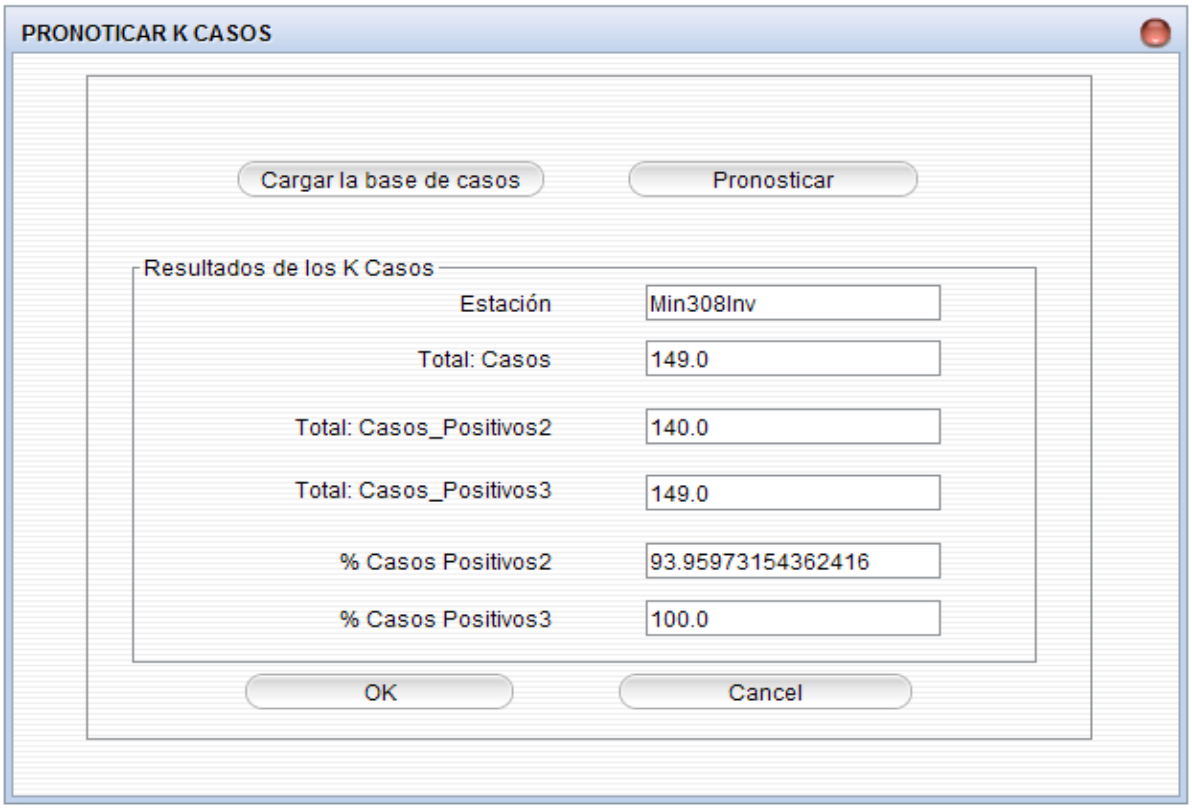

Figura 3.8 Ventana que muestra los resultados del pronóstico de los K-casos.

3.4 Resultados y Discusión

Validación de los Clasificadores

WEKA tiene implementado muchas facilidades de magnitudes de salidas y evaluación como la validación cruzada. En nuestro trabajo, se utilizó el método de validación cruzada con *k*-subconjuntos (k = 10 en nuestro caso), teniendo en cuenta que el tamaño de las bases de casos no son muy grandes y que este método nos permite utilizar cada caso tanto para entrenamiento como para prueba. A continuación se muestran los resultados obtenidos por las distintas configuraciones del los kernels del modelo PG. Se muestra el error medio absoluto y la desviación estándar de cada base de caso según el modelo.

Tabla 3.1 Resultados de la evaluación del pronóstico de temperatura mínima realizada mediante la configuración de los tres kernels del modelo PG para las 5 estaciones.

Estación	*PUK*	*PolyKernel*	*RBF*
308Inv	0.89(0.17) : Puk 1	1.16(0.23): PolyK 1	0.83(0.17): RBF 7
308Ver	0.70(0.10): Puk 9	0.94(0.28): PolyK2	0.73(0.19): RBF 8
326Inv	1.41(0.19): Puk 1	1.94(0.32): PolyK 8	1.39(0.19): RBF 9
326Ver	0.44(0.06): Puk 7, Puk 9	0.45(0.06): PolyK 7	0.44(0.06): RBF 8, RBF 9
338Inv	1.45(0.15): Puk 9	1.88(0.22) : PolyK 4	1.47(0.15): RBF 8
338Ver	0.89(0.12): Puk 9	1.09(0.16): PolyK 7	0.89(0.11): RBF 8
343Inv	1.01(0.12): Puk 1	1.47(0.21): PolyK 4	1.00(0.12): RBF 8
343Ver	0.75(0.15): Puk 1, Puk8	1.00(0.19): PolyK 2, PolyK 7	0.74(0.14): RBF 8
348Inv	0.99(0.13): Puk 9	1.33(0.18): PolyK 2	1.00(0.14): RBF 7
348Ver	0.79(0.17): Puk1	0.92(0.18): PolyK 7	0.79(0.17): RBF 7, RBF9

Tabla 3.2 Resultados de la evaluación del pronóstico de temperatura mínima realizada mediante IBk usando la medida Euclidiana con la distancia 1-distance para las 5 estaciones (Ver Anexo 1)

Estación	*K=3*	*K=4*	*K=5*	*K=6*	*K=8*	*K=9*	*K=18*
308Inv	1.32(0.31))	1.36(0.30)	1.39(0.30)	1.42(0.31)	1.45(0.33)	1.48(0.33)	1.62(0.36)
308Ver	0.92(0.15)	0.91(0.15)	0.90(0.15)	0.90(0.15)	0.89(0.16)	0.90(0.16)	0.98(0.17)
326Inv	2.03(0.30)	2.01(0.28)	2.01(0.28)	2.04(0.30)	2.06(0.29)	2.07(0.28)	2.21(0.28)
326Ver	0.45(0.06)	0.45(0.07)	0.45(0.07)	0.45(0.06)	0.45(0.06)	0.45(0.06)	0.47(0.06)
338Inv	1.84(0.27)	1.82(0.27)	1.86(0.26)	1.92(0.26)	1.95(0.26)	1.98(0.25)	2.12(0.25)
338Ver	1.01(0.15)	1.00(0.15)	1.00(0.15)	0.99(0.15)	1.00(0.15)	1.00(0.15)	1.05(0.15)
343Inv	1.50(0.23)	1.55(0.23)	1.59(0.23)	1.58(0.24)	1.58(0.25)	1.60(0.25)	1.74(0.22)
343Ver	0.99(0.18)	0.96(0.17)	0.93(0.16)	0.92(0.17)	0.91(0.17)	0.90(0.18)	0.89(0.18)
348Inv	1.33(0.16)	1.34(0.17)	1.35(0.18)	1.34(0.18)	1.36(0.19)	1.36(0.20)	1.44(0.20)
348Ver	0.96(0.20)	0.93(0.19)	0.91(0.19)	0.92(0.19)	0.91(0.19)	0.92(0.19)	0.96(0.18)

Tabla 3.3 Resultados de la evaluación del pronóstico de temperatura mínima realizada mediante IBk usando la distancia Euclidiana y la distancia pesada 1/distance para las 5 estaciones.

Estación	*K=3*	*K=4*	*K=5*	*K=6*	*K=8*	*K=9*	*K=18*
308Inv	1.29(0.30)	1.32(0.29)	1.35(0.29)	1.38(0.30)	1.40(0.32)	1.42(0.32)	1.54(0.35)
308Ver	0.91(0.15)	0.90(0.15)	0.89(0.15)	0.89(0.15)	0.87(0.16)	0.88(0.16)	0.93(0.16)
326Inv	2.00(0.29)	1.98(0.28)	1.96(0.28)	1.98(0.29)	2.00(0.28)	2.00(0.28)	2.12(0.28)
326Ver	0.45(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.45(0.06)
338Inv	1.82(0.26)	1.79(0.27)	1.82(0.26)	1.86(0.26)	1.89(0.25)	1.91(0.25)	2.03(0.25)
338Ver	0.99(0.15)	0.98(0.15)	0.97(0.14)	0.96(0.15)	0.97(0.14)	0.97(0.14)	1.00(0.15)
343Inv	1.46(0.23)	1.50(0.22)	1.53(0.23)	1.52(0.24)	1.53(0.25)	1.54(0.24)	1.65(0.23)
343Ver	0.96(0.18)	0.94(0.17)	0.91(0.16)	0.90(0.17)	0.89(0.17)	0.89(0.17)	0.87(0.18)
348Inv	1.30(0.16)	1.30(0.16)	1.31(0.17)	1.30(0.17)	1.31(0.18)	1.31(0.19)	1.38(0.20)
348Ver	0.95(0.20)	0.91(0.19)	0.90(0.19)	0.90(0.19)	0.89(0.19)	0.89(0.19)	0.92(0.18)

Tabla 3.4 Resultados al pronosticar con IBk usando Manhattan como medida y la distancia pesada 1-distance.

Estación	*K=3*	*K=4*	*K=5*	*K=6*	*K=8*	*K=9*	*K=18*
308Inv	1.29(0.30)	1.32(0.29)	1.35(0.29)	1.38(0.30)	1.40(0.32)	1.42(0.32)	1.54(0.35)
308Ver	0.91(0.15)	0.90(0.15)	0.89(0.15)	0.89(0.15)	0.87(0.16)	0.88(0.16)	0.93(0.16)
326Inv	2.00(0.29)	1.98(0.28)	1.96(0.28)	1.98(0.29)	2.00(0.28)	2.00(0.28)	2.12(0.28)
326Ver	0.45(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.44(0.06)	0.45(0.06)
338Inv	1.82(0.26)	1.79(0.27)	1.82(0.26)	1.86(0.26)	1.89(0.25)	1.91(0.25)	2.03(0.25)
338Ver	0.99(0.15)	0.98(0.15)	0.97(0.14)	0.96(0.15)	0.97(0.14)	0.97(0.14)	1.00(0.15)
343Inv	1.46(0.23)	1.50(0.22)	1.53(0.23)	1.52(0.24)	1.53(0.25)	1.54(0.24)	1.65(0.23)
343Ver	0.96(0.18)	0.94(0.17)	0.91(0.16)	0.90(0.17)	0.89(0.17)	0.89(0.17)	0.87(0.18)
348Inv	1.30(0.16)	1.30(0.16)	1.31(0.17)	1.30(0.17)	1.31(0.18)	1.31(0.19)	1.38(0.20)
348Ver	0.95(0.20)	0.91(0.19)	0.90(0.19)	0.90(0.19)	0.89(0.19)	0.89(0.19)	0.92(0.18)

Tabla 3.5 Resultados al pronosticar con IBk usando Manhattan como medida y la distancia pesada 1/distance.

Estación	*K=3*	*K=4*	*K=5*	*K=6*	*K=8*	*K=9*	*K=18*
308Inv	1.30(0.30)	1.31(0.29)	1.32(0.29)	1.34(0.30)	1.34(0.30)	1.34(0.31)	1.43(0.33)
308Ver	0.90(0.15)	0.86(0.14)	0.84(0.14)	0.86(0.14)	0.82(0.14)	0.82(0.14)	0.85(0.15)
326Inv	1.99(0.29)	1.96(0.28)	1.92(0.28)	1.90(0.28)	1.91(0.27)	1.91(0.27)	2.00(0.27)
326Ver	0.44(0.06)	0.43(0.06)	0.43(0.06)	0.43(0.06)	0.43(0.06)	0.43(0.06)	0.44(0.06)
338Inv	1.76(0.25)	1.74(0.25)	1.74(0.26)	1.77(0.24)	1.78(0.23)	1.80(0.24)	1.94(0.23)
338Ver	0.97(0.14)	0.96(0.14)	0.94(0.14)	0.94(0.14)	0.95(0.14)	0.95(0.14)	0.99(0.14)
343Inv	1.47(0.24)	1.48(0.24)	1.48(0.24)	1.46(0.24)	1.46(0.24)	1.46(0.24)	1.55(0.23)
343Ver	0.92(0.18)	0.89(0.17)	0.87(0.17)	0.87(0.17)	0.86(0.17)	0.86(0.17)	0.85(0.18)
348Inv	1.24(0.16)	1.24(0.17)	1.26(0.17)	1.26(0.17)	1.26(0.18)	1.25(0.18)	1.31(0.20)
348Ver	0.94(0.20)	0.92(0.20)	0.92(0.19)	0.91(0.19)	0.90(0.19)	0.89(0.19)	0.88(0.18)

En la Tabla 3.2 a la 3.5 se muestran los resultados alcanzados al pronosticar la temperatura con el modelo IBk con distintos valores de K y las medidas de distancias. En las dos primeras se utiliza la medida Euclidiana como método de búsqueda con la distancia pesada 1-distance y 1/distance respectivamente. En las Tabla 3.4 y Tabla 3.5 se muestran las mismas configuraciones pero con la medida Manhattan. Se puede ver que con este modelo no se obtiene mejor resultado que el anterior. Sin embargo se obtiene mejor resultado con la medida Manhattan comparado con las otras configuraciones del modelo IBk. A continuación se muestra los resultados obtenido con el modelo SVM usando distintas configuraciones del kernel Puk. Para este modelo solo empleamos el kernel Puk ya que hemos visto que es capaz de hacer pronóstico y que está probado que se puede emplear como un kernel universal para ahorrar el tiempo como se explicó en el Capítulo 2.

Tabla 3.7 Resultados al pronosticar con el modelo SVM y Puk como núcleo.

Estación	*Puk 11*	*Puk 12*	*Puk 13*	*Puk 14*	*Puk 15*
308Inv	1.37(0.26)	1.16(0.23)	1.17(0.27)	1.23(0.29)	1.35(0.33)
308Ver	1.25(0.18)	0.72(0.17)	1.38(0.22)	1.04(0.48)	1.25(1.02)
326Inv	2.58(0.33)	2.02(0.30)	1.51(0.26)	1.96(0.31)	2.11(0.32)
326Ver	0.46(0.06)	0.48(0.06)	0.44(0.06)	0.48(0.06)	0.47(0.07)
338Inv	1.52(0.21)	1.69(0.20)	1.53(0.21)	1.68(0.22)	1.54(0.23)
338Ver	0.90(0.12)	1.10(0.16)	1.00(0.18)	1.01(0.15)	0.92(0.12)
343Inv	1.20(0.20)	1.01(0.19)	1.03(0.19)	1.11(0.21)	1.24(0.27)
343Ver	0.88(0.17)	0.83(0.15)	0.88(0.17)	1.01(0.20)	1.18(0.24)
348Inv	1.0.4(0.13)	1.03(0.14)	1.33(0.19)	1.38(0.22)	1.49(0.26)
348Ver	0.97(0.17)	0.94(0.18)	0.79(0.18)	1.05(0.19)	1.15(0.21)

La Tabla 3.8 muestra las mejores configuraciones por cada modelo para las distintas bases de casos. Se puede observar que los kernels de base radiales RBF y el kernel Puk obtienen mejor resultado en base al error medio absoluto y la desviación estándar.

Tabla 3.8 Mejores configuraciones por cada modelo para las distintas bases de casos.

Estación	PG	MLP	IBK	SVM
308Inv	0.83(0.17): RBF 7	0.86(0.19): MLP2	1.29(0.30): K=3, E,-I	1.16(0.23): Puk12
308Ver	0.70(0.10): Puk 9	0.73(0.12): MLP 2,3	0.82(0.14): K=8, M,-I	0.72(0.28): Puk12
326Inv	1.39(0.19): RBF 9	1.41(0.19): MLP 2,3	1.90(0.28): K=6, M,-I	1.51(0.26): Puk 13
326Ver	0.44(0.06):Puk(7,9), RBF(8,9)	0.44(0.07): MLP 2	0.43 (0.06): K=6, M,-I	0.44(0.06): Puk 13
338Inv	1.45(0.15): Puk 9	1.59 (0.16): MLP 2	1.74(0.26): K= 4,M,-I	1.52(0.21): Puk 11
338Ver	0.89(0.11): RBF 8	0.95(0.12): MLP 2	0.94(0.14): K=5, M, -I	0.90(0.12): Puk 11
343Inv	1.00(0.12): RBF 8	1.05(0.14): MLP 2	1.46 (0.24): K =8,M, -I	1.01(0.19): Puk 12
343Ver	0.74(0.14): RBF 8	0.84(0.18): MLP 2	0.86 (0.17): K =8,M, -I	0.83(0.15): Puk12
348Inv	0.99(0.13): Puk 9	1.05(0.16): MLP 2,3	1.24 (0.17): K=4,M,-I	1.04(0.13): Puk 11
348Ver	0.79(0.17): Puk1, RBF 7,9	0.81(0.18): MLP 2,3	0.88(0.18): K=18,M,-I	0.79(0.18): Puk13

3.5 Comparación de los resultados de los modelos de SAGI con los del modelo MOS

Se evaluó la efectividad de los modelos en "% de acierto", tomando como umbral de precisión el valor de $\pm 2^{0}C$, establecido en la actualidad en la "Metodología de evaluación de los pronósticos", del Sistema Nacional de Pronósticos del Instituto de Meteorología de Cuba. Finalmente se realizó una comparación de los resultados generales del modelo que se presenta con los que se obtienen directamente del modelo global GFS. Se calculan los errores para cada instancia de la base de casos y se dice que los casos son positivos si el error de cada instancia está en el rango explicado previamente, o negativos en caso contrario. A continuación se puede observar los resultados obtenidos por los dos modelos, MOS y el SAGI.

Tabla 3.9 Resultados de la evaluación del pronóstico de las temperaturas minímas realizada para las 5 estaciones para el período Marzo de 2011 con ayuda del modelo MOS en el CMPVC.

Estación	Casos Total	Casos Positivos	%Casos Positivos	Error Medio Absoluto
308Inv	23	20	86.9565217391304	1.26650391304348
326Inv	31	24	77.4193548387097	1.38805451612903
338Inv	31	17	54.8387096774194	2.22155432258064
343Inv	31	26	83.8709677419355	0.968560290322581
348Inv	31	24	77.4193548387097	1.34960206451613
Total	147	11	76.10098176718094	1.438855021318372

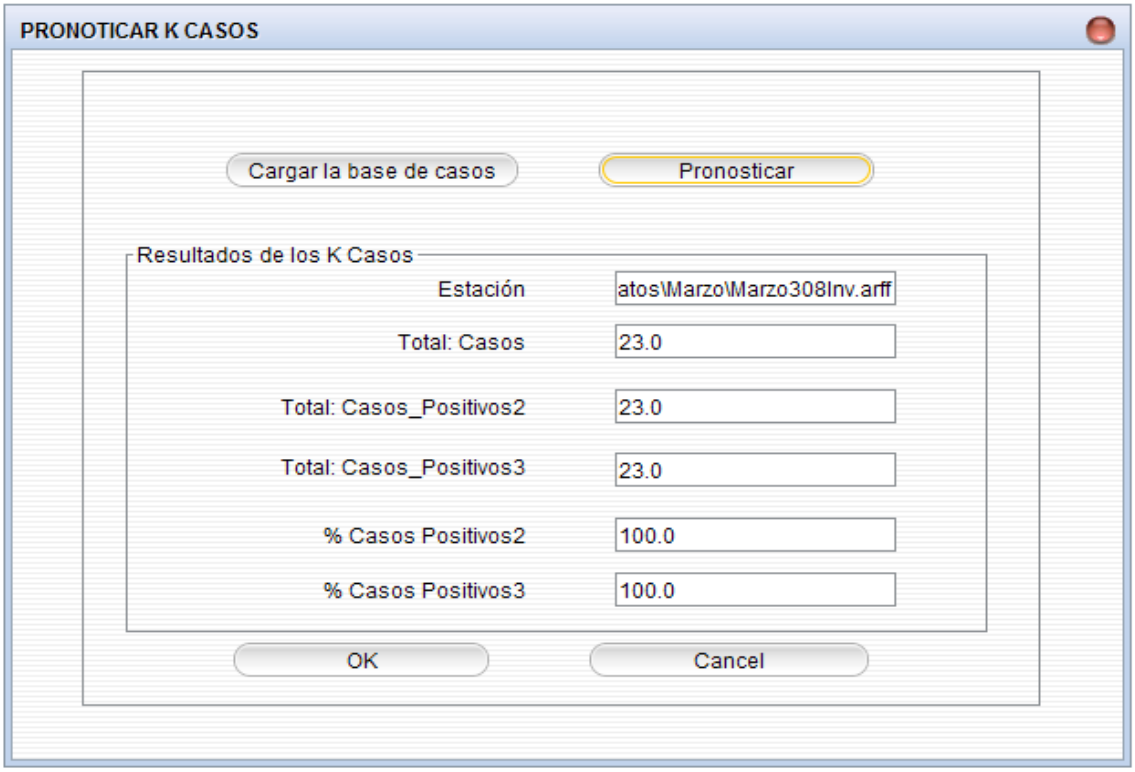

Figura 3.9 Resultados de la evaluación del pronóstico de temperatura mínima realizada para la estación 308Inv para el período Marzo de 2011 usando el modelo SAGI

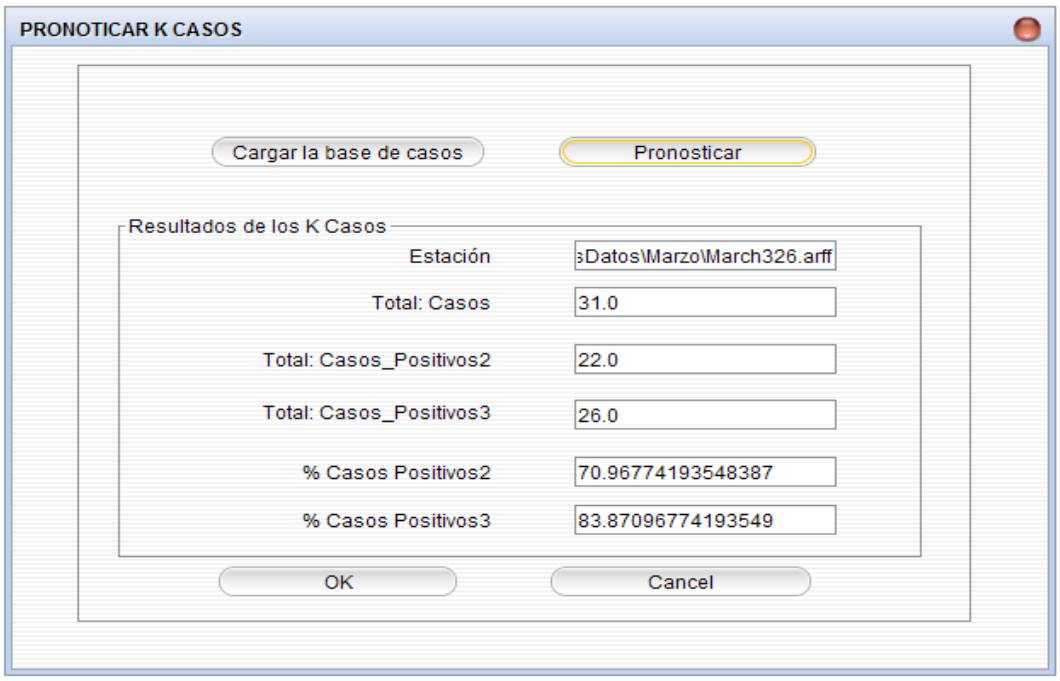

Figura 3.10 Resultados de la evaluación del pronóstico de temperatura mínima realizada para la estación 326Inv para el período Marzo de 2011 usando el modelo SAGI.

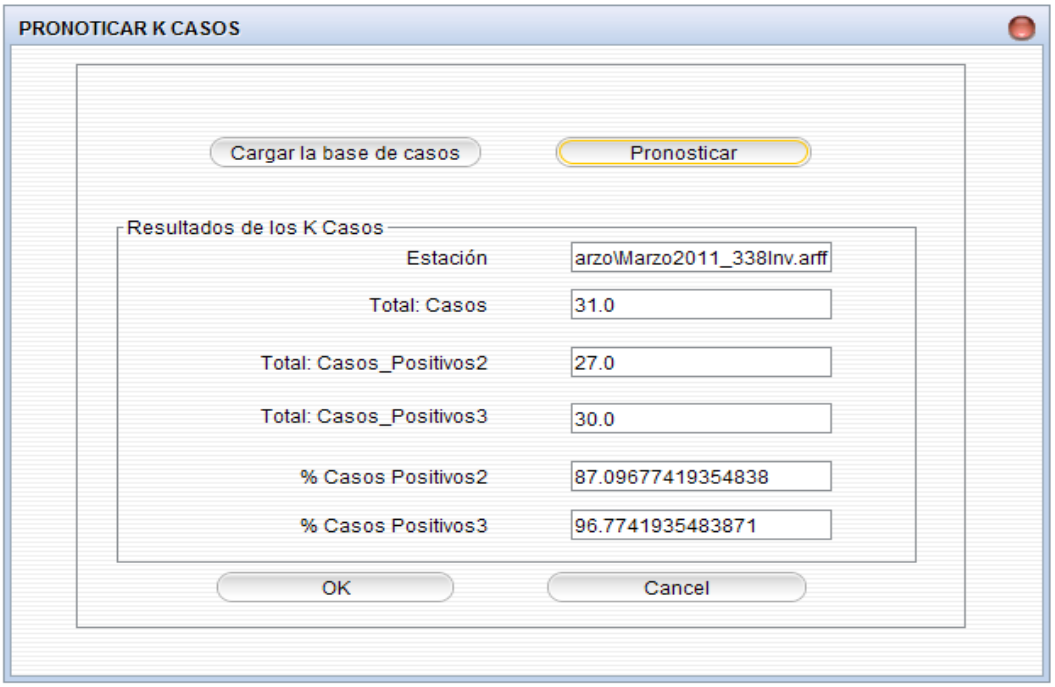

Figura 3.11 Resultados de la evaluación del pronóstico de temperatura mínima realizada para la estación 338Inv para el período Marzo de 2011 usando el modelo SAGI.

Tabla 3.12 Resultados de la evaluación del pronóstico de temperatura mínima realizada para las estaciones para el período Marzo de 2011 mediante SAGI (duda, por qué no la misma cantidad de casos totales).

Estación	Casos Total	Casos Positivos	%Casos Positivos	Error Medio Absoluto
308Inv	23	23	100	0.664164270225191
326Inv	31	22	70.96774193548387	1.6621376077237202
338Inv	31	27	87.09677419354838	1.8348512762259364
343Inv	31	29	93.54838709677419	0.9169166795796209
348Inv	31	31	100	0.9153543893449123
Total	147	132	90.32258064516129	1.015301508703952

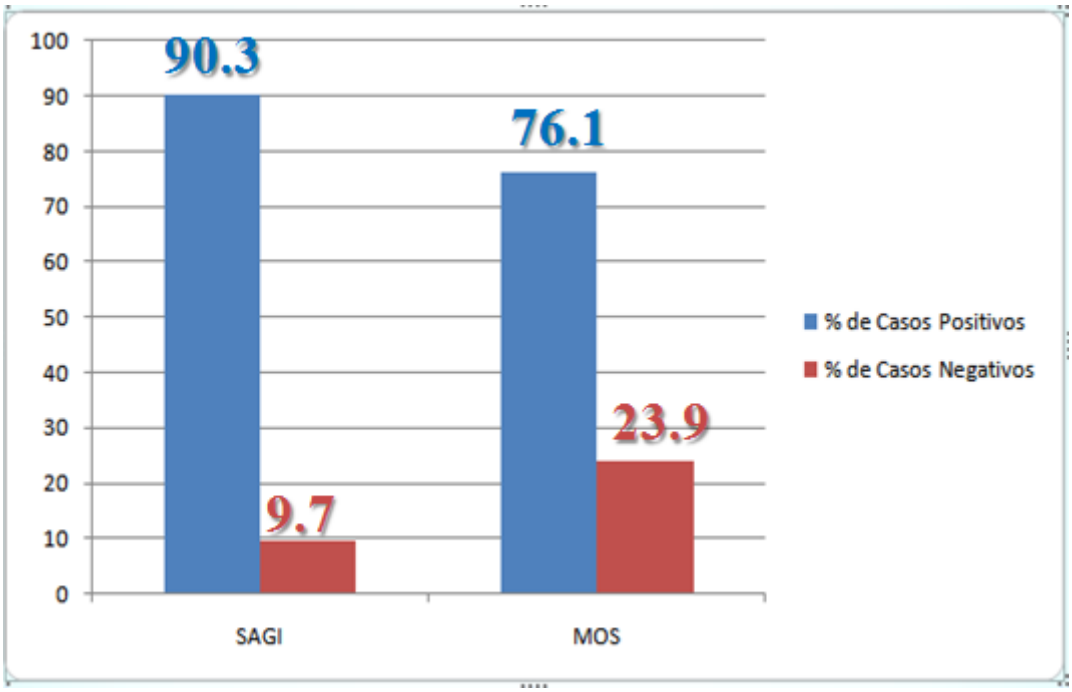

Figura 3.12 Comparación de los modelo SAGI y MOS según el porciento de acierto realizada para las 5 estaciones para el período Marzo de 2011

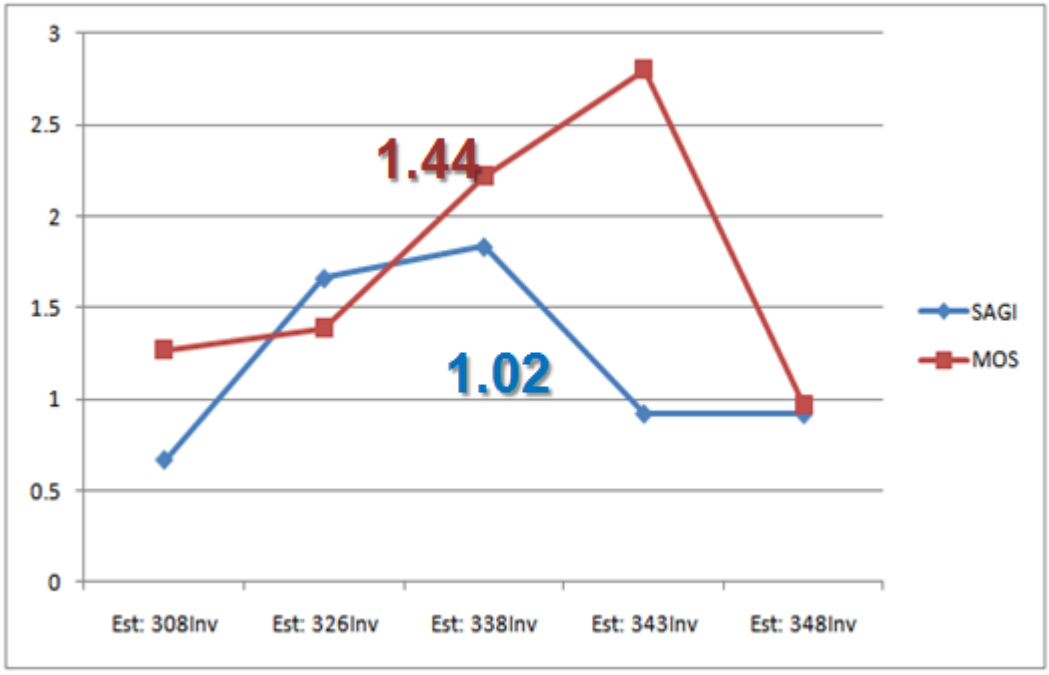

Figura 3.13 Comparación de los modelo SAGI y MOS según el porciento de acierto realizada para las 5 estaciones para el período Marzo de 2011

Tabla 3.14 Las configuraciones de los modelo de la aplicación SAGI empleados para el pronóstico de las temperaturas mínimas para las 5 estaciones para el periodo invierno (Marzo 2011).

343Inv	RBF 8	Noise = 0.01	Gamma= 0.0008	-
308Inv	RBF 7	Noise = 0.08	Gamma= 0.0006	-
326Inv	RBF 9	Noise = 0.006	Gamma = 0.008	-
338Inv	Puk 9	Noise = 0.2	Omega = 0.6	Sigma = 4.5
348Inv	Puk 9	Noise = 0.2	Omega = 0.6	Sigma = 4.5

En la Tabla 3.9 se observa que la efectividad del modelo MOS fue de 76.1 % en promedio para los casos positivos, mientras que fue 90.3% con los modelos de la aplicación SAGI (ver Tabla 3.10). En tanto al error medio absoluto resultó de 1.44 en promedio para el MOS mientras que se obtuvieron el error de 1.02 con la aplicación SAGI. También se observa que los modelos usados en la aplicación SAGI superan el modelo MOS para todas estaciones excepto para la estación 326Inv donde el MOS supero los modelos de la aplicación SAGI en tanto al porciento de acierto que en el error medio absoluto. Para esta estación los resultados mediante SAGI alcanzan un 70.97 por ciento de acierto con un error de 1.66 mientras que el MOS alcanzo un 77.4 de acierto con un error de 1.388. Los resultados se puede apreciar bien en la Figura 2.4 a

la Figura 2.5. En general los resultados alcanzados con las nuevas técnicas superan a los alcanzados por el modelo MOS.

Los resultados reportados proporcionan una guía de cómo las técnicas de aprendizaje automatizado pueden ser una herramienta útil para hacer predicciones en la vida real.

3.3 Conclusiones parciales del capítulo

Este capítulo ha presentado una aplicación de los cuatro métodos de aprendizaje automatizado (SVM, PG, RNN y KNN) para regresión, específicamente al problema del pronóstico de temperaturas mínimas. Se mostró a través del estudio experimental con 10 bases de casos obtenido del CMPVC el buen desempeño de los modelos propuestos en este trabajo. También se incluyó una comparación de estos resultados con el modelo MOS y de esta comparación se destaca que las técnicas propuestas en este trabajo se pueden utilizar para conseguir una mejor predicción para este tipo de datos en los que existe una relación compleja. En cuanto al porciento de los casos positivos obtenidos por las nuevas técnicas, la capacidad predictiva del modelo SAGI resultó mejor para 4 de las 5 estaciones utilizadas en el trabajo para el periodo Marzo 2011.

Conclusiones

En el trabajo, se utilizaron datos de las temperaturas mínimas de 5 estaciones de la provincia de Villa Clara para el tiempo invierno y verano. Se muestra cómo diferentes técnicas de aprendizaje automatizado generan resultados de predicción numérica potencialmente competitivos.

El análisis experimental realizado con 10 conjuntos de datos de las 5 estaciones de la provincia de Villa Clara permite contar con un modelo de AA para predecir las temperaturas mínimas.

Se implementó la interfaz de entrada de datos la cual permite el entrenamiento con los modelos de AA y es posible pronosticar nuevos casos de una estación dada.

Se compararon los resultados alcanzados por los modelos incorporados en la aplicación SAGI con los alcanzados por el modelo MOS y se concluye que las técnicas propuestas en este trabajo se pueden utilizar para conseguir una mejor predicción para pronosticar las temperaturas mínimas en el CMPVC.

Recomendaciones

Los resultados alcanzados, así como el estudio de los parámetros de los modelos realizados permitieron formular nuevas tareas de investigación, las cuales quedan formuladas como recomendaciones del presente trabajo y son:

1. Culminar el análisis experimental para el resto de las estaciones meteorológicas que atiende el CMPVC.
2. Añadir una nueva funcionalidad a SAGI de modo que en la etapa de entrenamiento se proponga la mejor configuración para cada una de las técnicas de AA para cada conjunto de datos.
3. Emplear la combinación de clasificadores simples o sea emplear los meta-clasificadores con el fin de mejorar los resultados del pronóstico de la temperaturas mínimas.

Referencias bibliográficas

1. Liu, K.F.-R. and C.-W. Yu, *Integrating case-based and fuzzy reasoning to qualitatively predict risk in an environmental impact assessment review.* Environmental Modelling & Software, 2009. **24**(10): p. 1241-1251.
2. Ellis, R.H., Hadley, P., Roberts, E.H. and Summerfield, R.J. (1990) *Quantitative relations between temperature and crop development and growth.*, 85-115.
3. Ellis, R.H., Hadley, P., Roberts, E.H. and Summerfield, R.J. (1990) *Quantitative relations between temperature and crop development and growth.* . Climate Change and Genetic Resources, 85-115.
4. Gamble, J.L.e., K.L. Ebi, F.G. Sussman, T.J. Wilbanks, (2008) *Analyses of the effects of global change on human health and welfare and human systems* U.S. Climate Change Science Program and the Subcommittee on Global Change Research. .
5. Glahn, H.R. and D.A. Lowry, *The use of model output statistics (MOS) in objective weather forecasting.* Journal of applied meteorology, 1972. **11**(8): p. 1203-1211.
6. Moreno, A.E., *Aplicación de métodos matemáticos para un modelo numérico de pronóstico del tiempo.*
7. Maduna, N., *Short-Term Local Weather Forecasting Using Model Output Statistics.* 2007.
8. Macy, A.S.P.R.B., *Pattern Recognition with Neural Networks in C++* 1995, CRC Press, CRC Press LLC
9. Zheng, L. and X. He, *Classification techniques in pattern recognition.* 2005.
10. Castillo, E.R., *Máquina de Soporte Vectorial en el Análisis de Serie de Tiempo*, in *Campus Montecillo*. 2012, Institución de Enseñanza e Investigación en Ciencias Agrícolas: Montecillo.
11. Rao, T., N. Rajasekhar, and T. Rajinikan, *An efficient approach for Weather forecasting using Support Vector Machines.* International Proceedings of Computer Science & Information Technology, 2012. **47**.
12. Martín, J.C., *Predicción de eventos deportivos empleando procesos gaussianos: Tenis.*, in *Departamento de Teoría de la señal y comunicaciones*. 2011.
13. Ian H. Witten, E.F., Mark A. Hall, *Data mining : practical machine learning tools and techniques*. 2011, Morgan Kaufmann: Burlington.

14. et.al, R.R., *WEKA Manual for Version 3-6-2*. 2010, University of Waikato, Hamilton, New Zealand.
15. Ruiz, R., *Heurísticas de selección de atributos para datos de gran dimensionalidad*, in *Departamento de Lenguajes y Sistemas Informáicos*. 2006, Universidad de Sevilla: Sevilla.
16. Pierre Baldi, S.B., *Bioinformatics: The Machine Learning Approach*, D.H. Christopher Bishop, Michael Jordan, and Michael Kearns, Editor. 2001: London, England.
17. Bonifacio Martín del Brío and A.S. Molina, *Redes Neuronales y Sistemas Difusos*. 2001: Zaragoza, España
18. Matich, D.J., *Redes Neuronales: Conceptos básicos y aplicaciones.* Cátedra de Informática Aplicada a la Ingeniería de Procesos–Orientación I, 2001.
19. Perez, R.B., *Curso Introductorio a las Redes Neuoronales Artificiales.* 1993.
20. Bello Pérez, R., *Curso Introductorio de Redes Neuronales Artificiales.* McGrawHill, Madrid, España, 1993.
21. Altan Dombaycı, Ö. and M. Gölcü, *Daily means ambient temperature prediction using artificial neural network method: A case study of Turkey.* Renewable Energy, 2009. **34**(4): p. 1158-1161.
22. Baboo, D.S.S. and I.K. Shereef, *An Efficient Temperature Prediction System using BPN Neural Network.* International journal of environmental science and development, 2011. **2**(1): p. 49-54.
23. Baboo, S.S. and I.K. Shereef, *An efficient weather forecasting system using artificial neural network.* International journal of environmental science and development, 2010. **1**(4): p. 2010-0264.
24. Bonilla, J.E., J. Ramírez & O. Ramírez (2006) *A Metodology of design for a One-variable Neural Network model to forecast the minimum temperature on the Mosquera zone*.
25. Hall, T., H.E. Brooks, and C.A. Doswell III, *Precipitation forecasting using a neural network.* Weather and forecasting, 1999. **14**(3): p. 338-345.
26. Hayati, M. and Z. Mohebi, *Application of artificial neural networks for temperature forecasting.* World Academy of Science, Engineering and Technology, 2007. **28**(2): p. 275-279.

27. Sharma, A. and S. Agarwal, *Temperature prediction using wavelet neural network.* Res. J. Inform. Technol, 2012. **4**: p. 22-30.
28. Pal, N.R., et al., *SOFM-MLP: A hybrid neural network for atmospheric temperature prediction.* Geoscience and Remote Sensing, IEEE Transactions on, 2003. **41**(12): p. 2783-2791.
29. Minns, A.W. *Extended rainfall-runoff modelling using artificial neural networks*. in *Hydroinformatics*. 1996.
30. Kuligowski, R.J. and A.P. Barros, *Experiments in short-term precipitation forecasting using artificial neural networks.* Monthly weather review, 1998. **126**(2): p. 470-482.
31. Khotanzad, A., et al., *An artificial neural network hourly temperature forecaster with applications in load forecasting*. Power Systems, IEEE Transactions on, 1996. **11**(2): p. 870-876.
32. Kadu, P.P., Wagh, Kishor P, Chatur, Prashant N, *Temperature Prediction System Using Back propagation Neural Network: An Approch.* International Journal of Computer Science & Communication Networks. **2(1)**.
33. Aldo S. Moya , A.E., José M. Ortega , Arisleydis Peña, *Modelo físico estadístico de pronóstico de temperaturas extremas para Cuba.*
34. Schildt, H., *Java: A bigginers's guide, Third Edition*. 2005, The McGraw-Hill.
35. Linoff, M.J.A.B.a.G.S., ed. *Data Mining Techniques*. Second Edition ed. 2004.
36. B. Ustun, W.J.M., L.M.C. Buydens, *Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel.* ELSEVIER, 2005.

Anexos

Anexo 1: Resultados de los estudios de parámetros realizados para los cuatros modelos propuestos.

```
Resultsets: 10
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 1:17 AM

Dataset                   (1) functions.Gau | (2) functions. (3) functions. (4) functions. (5) functions. (6) functions. (7) functions. (8) functions. (9) functions.
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
Min08Inv                  (100)   0.89(0.17) |   2.05(0.42) v   2.15(0.43) v   1.87(0.38) v   1.05(0.25) v   1.19(0.29) v   0.91(0.18)     0.91(0.18) v   0.90(0.19)
Min08Ver                  (100)   0.82(0.13) |   1.20(0.17) v   1.27(0.17) v   1.25(0.16) v   0.77(0.13)     0.80(0.14)     0.79(0.12) *   0.80(0.12) *   0.70(0.10) *
Min26Inv                  (100)   1.41(0.19) |   2.89(0.33) v   3.09(0.34) v   2.57(0.29) v   1.58(0.22) v   1.66(0.22) v   1.42(0.19)     1.42(0.19)     1.41(0.20)
Min26Ver                  (100)   0.46(0.06) |   0.46(0.06)     0.51(0.06) v   0.45(0.06) *   0.45(0.06) *   0.47(0.06) v   0.45(0.06) *   0.44(0.06) *   0.44(0.06) *
Min38Inv                  (100)   1.53(0.15) |   2.53(0.27) v   2.68(0.28) v   2.32(0.25) v   1.51(0.17)     1.61(0.18)     1.51(0.15)     1.51(0.15)     1.45(0.15) *
Min38Ver                  (100)   0.91(0.11) |   1.39(0.20) v   1.58(0.22) v   1.35(0.19) v   0.91(0.13)     0.90(0.13)     0.90(0.12)     0.90(0.12)     0.89(0.12)
Min43Inv                  (100)   1.01(0.12) |   2.40(0.25) v   2.60(0.26) v   2.13(0.22) v   1.14(0.14) v   1.22(0.16) v   1.02(0.13)     1.02(0.12)     1.01(0.13)
Min43Ver                  (100)   0.75(0.15) |   1.23(0.20) v   1.42(0.20) v   1.14(0.19) v   0.85(0.16) v   0.81(0.16) v   0.75(0.15)     0.75(0.15)     0.77(0.15)
Min48Inv                  (100)   1.00(0.14) |   1.98(0.23) v   2.11(0.24) v   1.73(0.21) v   1.06(0.14)     1.11(0.15) v   1.00(0.13)     1.01(0.13)     0.99(0.13)
Min48Ver                  (100)   0.79(0.17) |   1.14(0.19) v   1.24(0.19) v   1.07(0.18) v   0.90(0.18) v   0.85(0.18) v   0.81(0.17) v   0.80(0.17) v   0.85(0.17) v
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
                                    (v/ /*) |     (9/0/0)     (9/0/0)     (9/0/0)     (4/5/0)     (6/3/0)     (1/7/1)     (2/6/1)     (1/6/2)

Key:
(1) functions.GaussianProcesses '-L 0.4 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 13.0 -S 4.2\"' -8620066949967678545
(2) functions.GaussianProcesses '-L 3.5 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.1 -S 14.5\"' -8620066949967678545
(3) functions.GaussianProcesses '-L 3.5 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.4 -S 14.5\"' -8620066949967678545
(4) functions.GaussianProcesses '-L 1.0 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 1.0 -S 15.5\"' -8620066949967678545
(5) functions.GaussianProcesses '-L 0.4 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 1.0 -S 1.0\"' -8620066949967678545
(6) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.4 -S 0.8\"' -8620066949967678545
(7) functions.GaussianProcesses '-L 0.4 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.5 -S 5.5\"' -8620066949967678545
(8) functions.GaussianProcesses '-L 0.4 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.6 -S 5.5\"' -8620066949967678545
(9) functions.GaussianProcesses '-L 0.2 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 0.6 -S 4.5\"' -8620066949967678545
(10) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.Puk -C 250007 -O 2.5 -S 3.5\"' -8620066949967678545
```

```
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 10
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 3:59 AM


Dataset                 (1) functions.Gau | (2) functions.  (3) functions.  (4) functions.  (5) functions.  (6) functions.  (7) functions.  (8) functions.
-------------------------------------------------------------------------------------------------------------------------------------------------------------
Min308Inv               (100)   1.37(0.26) |   1.16(0.23) *   1.17(0.27) *   1.23(0.29)     1.35(0.33)     1.36(0.26)     1.17(0.23) *   1.19(0.28)
Min308Ver               (100)   1.25(0.18) |   0.94(0.28) *   0.94(0.36) *   1.04(0.48)     1.25(1.02)     1.24(0.19)     0.95(0.37) *   1.02(0.54)
Min326Inv               (100)   2.58(0.33) |   2.02(0.30) *   1.94(0.32) *   1.95(0.31) *   2.11(0.32) *   2.58(0.33)     2.01(0.30) *   1.94(0.32) *
Min326Ver               (100)   0.48(0.06) |   0.45(0.06) *   0.46(0.06) *   0.47(0.06)     0.47(0.07)     0.48(0.06)     0.45(0.06) *   0.46(0.06)
Min338Inv               (100)   2.22(0.26) |   1.99(0.20) *   1.92(0.21) *   1.88(0.22) *   1.94(0.23) *   2.22(0.26)     1.99(0.20) *   1.91(0.22) *
Min338Ver               (100)   1.29(0.18) |   1.10(0.16) *   1.14(0.18) *   1.17(0.18) *   1.24(0.20)     1.29(0.18)     1.09(0.16) *   1.14(0.17) *
Min343Inv               (100)   1.98(0.25) |   1.57(0.20) *   1.50(0.19) *   1.47(0.21) *   1.58(0.24) *   1.98(0.25)     1.56(0.19) *   1.48(0.19) *
Min343Ver               (100)   1.20(0.20) |   1.00(0.19) *   1.03(0.19) *   1.11(0.21)     1.24(0.27)     1.21(0.20) v   1.00(0.19) *   1.05(0.19) *
Min348Inv               (100)   1.40(0.20) |   1.33(0.18)     1.33(0.19)     1.38(0.22)     1.49(0.26)     1.40(0.20)     1.34(0.18)     1.33(0.19)
Min348Ver               (100)   0.97(0.17) |   0.92(0.18) *   0.98(0.19)     1.05(0.19) v   1.15(0.21) v   0.97(0.17)     0.92(0.18)     0.99(0.19)
-------------------------------------------------------------------------------------------------------------------------------------------------------------
                                 (v/ /*) |         (0/1/9)        (0/2/8)        (1/5/4)        (1/6/3)        (1/9/0)        (0/2/8)        (0/5/5)


Key:
(1) functions.GaussianProcesses '-L 0.8 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 1.0\"' -8620066949967678545
(2) functions.GaussianProcesses '-L 0.8 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 2.0\"' -8620066949967678545
(3) functions.GaussianProcesses '-L 0.8 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 3.0\"' -8620066949967678545
(4) functions.GaussianProcesses '-L 0.8 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 4.0\"' -8620066949967678545
(5) functions.GaussianProcesses '-L 0.8 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 5.0\"' -8620066949967678545
(6) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 1.0\"' -8620066949967678545
(7) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 2.0\"' -8620066949967678545
(8) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 3.0\"' -8620066949967678545
(9) functions.GaussianProcesses '-L 0.6 -N 0 -K \"functions.supportVector.PolyKernel -C 250007 -E 4.0\"' -8620066949967678545
```

```
Datasets:   9
Resultsets: 10
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 3:12 AM


Dataset                 (1) functions.Gau | (2) functions. (3) functions. (4) functions. (5) functions. (6) functions. (7) functions. (8) functions. (9) functions. (10) functions
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Min308Inv               (100)   1.08(0.22) |   1.05(0.21) *   2.20(0.43) v   0.91(0.18) *   0.94(0.19) *   1.32(0.28) v   0.83(0.18) *   0.90(0.19) *   0.83(0.17) *   0.87(0.17) *
Min308Ver               (100)   1.15(0.15) |   1.15(0.16)     1.27(0.17) v   1.09(0.18) *   1.11(0.18)     1.21(0.14) v   0.93(0.23) *   0.73(0.19) *   0.89(0.20) *   1.03(0.18) *
Min326Inv               (100)   1.54(0.19) |   1.52(0.19) *   3.18(0.35) v   1.44(0.19) *   1.46(0.19) *   1.76(0.20) v   1.40(0.19) *   1.40(0.20) *   1.39(0.19) *   1.42(0.19) *
Min326Ver               (100)   0.46(0.06) |   0.46(0.06)     0.51(0.06) v   0.45(0.06) *   0.45(0.06) *   0.47(0.06) v   0.45(0.06) *   0.44(0.06) *   0.44(0.06) *   0.45(0.06) *
Min338Inv               (100)   1.65(0.16) |   1.64(0.16) *   2.75(0.29) v   1.59(0.15) *   1.60(0.15) *   1.78(0.18) v   1.56(0.14) *   1.47(0.15) *   1.54(0.14) *   1.57(0.15) *
Min338Ver               (100)   1.02(0.13) |   1.01(0.13) *   1.65(0.23) v   0.97(0.12) *   0.99(0.12) *   1.09(0.15) v   0.95(0.11) *   0.89(0.11) *   0.93(0.11) *   0.96(0.12) *
Min343Inv               (100)   1.17(0.12) |   1.15(0.12) *   2.68(0.27) v   1.07(0.12) *   1.09(0.12) *   1.37(0.14) v   1.02(0.12) *   1.00(0.12) *   1.01(0.12) *   1.04(0.12) *
Min343Ver               (100)   0.82(0.15) |   0.82(0.15)     1.50(0.20) v   0.81(0.15) *   0.81(0.15) *   0.87(0.16) v   0.80(0.15)     0.74(0.14) *   0.78(0.15) *   0.80(0.15) *
Min348Inv               (100)   1.09(0.15) |   1.08(0.15) *   2.17(0.25) v   1.03(0.14) *   1.04(0.14) *   1.20(0.16) v   1.00(0.14) *   1.00(0.14) *   1.00(0.14) *   1.01(0.14) *
Min348Ver               (100)   0.82(0.17) |   0.82(0.17) *   1.29(0.19) v   0.80(0.17) *   0.80(0.17) *   0.87(0.17) v   0.79(0.18)     0.83(0.17)     0.79(0.18)     0.79(0.17) *
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------
                                  (v/ /*) |        (0/2/7)        (9/0/0)        (0/0/9)        (0/1/8)        (9/0/0)        (0/2/7)        (0/1/8)        (0/1/8)        (0/0/9)


Key:
(1) functions.GaussianProcesses '-L 0.2 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 0.008\"' -8620066949967678545
(2) functions.GaussianProcesses '-L 0.06 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 8.0E-4\"' -8620066949967678545
(3) functions.GaussianProcesses '-L 1.0 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 0.001\"' -8620066949967678545
(4) functions.GaussianProcesses '-L 0.04 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 8.0E-4\"' -8620066949967678545
(5) functions.GaussianProcesses '-L 0.04 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 6.0E-4\"' -8620066949967678545
(6) functions.GaussianProcesses '-L 0.08 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 6.0E-4\"' -8620066949967678545
(7) functions.GaussianProcesses '-L 0.01 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 8.0E-4\"' -8620066949967678545
(8) functions.GaussianProcesses '-L 0.006 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 0.008\"' -8620066949967678545
(9) functions.GaussianProcesses '-L 0.04 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 0.008\"' -8620066949967678545
(10) functions.GaussianProcesses '-L 0.1 -N 0 -K \"functions.supportVector.RBFKernel -C 250007 -G 0.008\"' -8620066949967678545
```

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 4:21 AM


Dataset                   (1) lazy.IBk '-K  | (2) lazy.IBk ' (3) lazy.IBk ' (4) lazy.IBk ' (5) lazy.IBk ' (6) lazy.IBk ' (7) lazy.IBk
-------------------------------------------------------------------------------------------------------------------------------------
Min308Inv                 (100)   1.32(0.31) |   1.36(0.30)     1.39(0.30)     1.42(0.31)     1.45(0.33)     1.48(0.33) v   1.62(0.36)
Min308Ver                 (100)   0.92(0.15) |   0.91(0.15)     0.90(0.15)     0.90(0.15)     0.89(0.16)     0.90(0.16)     0.98(0.17)
Min326Inv                 (100)   2.03(0.30) |   2.01(0.28)     2.01(0.28)     2.04(0.30)     2.06(0.29)     2.07(0.28)     2.21(0.28)
Min326Ver                 (100)   0.45(0.06) |   0.45(0.07)     0.45(0.07)     0.45(0.06)     0.45(0.06)     0.45(0.06)     0.47(0.06)
Min338Inv                 (100)   1.84(0.27) |   1.82(0.27)     1.86(0.26)     1.92(0.26)     1.95(0.26)     1.98(0.25) v   2.12(0.25)
Min338Ver                 (100)   1.01(0.15) |   1.00(0.15)     1.00(0.15)     0.99(0.15)     1.00(0.15)     1.00(0.15)     1.05(0.15)
Min343Inv                 (100)   1.50(0.23) |   1.55(0.23)     1.59(0.23) v   1.58(0.24)     1.58(0.25)     1.60(0.25)     1.74(0.22)
Min343Ver                 (100)   0.99(0.18) |   0.96(0.17)     0.93(0.16) *   0.92(0.17)     0.91(0.17)     0.90(0.18)     0.89(0.18)
Min348Inv                 (100)   1.33(0.16) |   1.34(0.17)     1.35(0.18)     1.34(0.18)     1.36(0.19)     1.36(0.20)     1.44(0.20)
Min348Ver                 (100)   0.96(0.20) |   0.93(0.19)     0.91(0.19)     0.92(0.19)     0.91(0.19)     0.92(0.19)     0.96(0.18)
-------------------------------------------------------------------------------------------------------------------------------------
                                    (v/ /*) |       (0/10/0)       (1/8/1)       (0/10/0)       (0/10/0)       (2/8/0)        (5/4/


Key:
(1) lazy.IBk '-K 3 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(2) lazy.IBk '-K 4 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(3) lazy.IBk '-K 5 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(4) lazy.IBk '-K 6 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(5) lazy.IBk '-K 8 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(6) lazy.IBk '-K 9 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(7) lazy.IBk '-K 18 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30
```

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 4:14 AM


Dataset                   (1) lazy.IBk '-K  | (2) lazy.IBk ' (3) lazy.IBk ' (4) lazy.IBk ' (5) lazy.IBk ' (6) lazy.IBk ' (7) lazy.IBk
-------------------------------------------------------------------------------------------------------------------------------------------
Min308Inv                 (100)   1.29(0.30) |   1.32(0.29)     1.35(0.29)     1.38(0.30)     1.40(0.32)     1.42(0.32) v   1.54(0.35)
Min308Ver                 (100)   0.91(0.15) |   0.90(0.15)     0.89(0.15)     0.89(0.15)     0.87(0.16)     0.88(0.16)     0.93(0.16)
Min326Inv                 (100)   2.00(0.29) |   1.98(0.28)     1.96(0.28)     1.98(0.29)     2.00(0.28)     2.00(0.28)     2.12(0.28)
Min326Ver                 (100)   0.45(0.06) |   0.44(0.06)     0.44(0.06)     0.44(0.06)     0.44(0.06)     0.44(0.06)     0.45(0.06)
Min338Inv                 (100)   1.82(0.26) |   1.79(0.27)     1.82(0.26)     1.86(0.26)     1.89(0.25)     1.91(0.25)     2.03(0.25)
Min338Ver                 (100)   0.99(0.15) |   0.98(0.15)     0.97(0.14)     0.96(0.15)     0.97(0.14)     0.97(0.14)     1.00(0.15)
Min343Inv                 (100)   1.46(0.23) |   1.50(0.22)     1.53(0.23)     1.52(0.24)     1.53(0.25)     1.54(0.24)     1.65(0.23)
Min343Ver                 (100)   0.96(0.18) |   0.94(0.17)     0.91(0.16) *   0.90(0.17) *   0.89(0.17)     0.89(0.17)     0.87(0.18)
Min348Inv                 (100)   1.30(0.16) |   1.30(0.16)     1.31(0.17)     1.30(0.17)     1.31(0.18)     1.31(0.19)     1.38(0.20)
Min348Ver                 (100)   0.95(0.20) |   0.91(0.19) *   0.90(0.19) *   0.90(0.19)     0.89(0.19)     0.89(0.19)     0.92(0.18)
-------------------------------------------------------------------------------------------------------------------------------------------
                              (v/ /*) |        (0/9/1)        (0/8/2)        (0/9/1)        (0/10/0)        (1/9/0)        (3/6/1


Key:
(1) lazy.IBk '-K 3 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(2) lazy.IBk '-K 4 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(3) lazy.IBk '-K 5 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(4) lazy.IBk '-K 6 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(5) lazy.IBk '-K 8 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(6) lazy.IBk '-K 9 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308
(7) lazy.IBk '-K 18 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30
```

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 4:17 AM


Dataset                  (1) lazy.IBk '-K  | (2) lazy.IBk ' (3) lazy.IBk ' (4) lazy.IBk ' (5) lazy.IBk ' (6) lazy.IBk ' (7) lazy.IBk '
-------------------------------------------------------------------------------------------------------------------------------------------
Min308Inv                (100)   1.30(0.30) |   1.31(0.29)     1.32(0.29)     1.34(0.30)     1.34(0.30)     1.34(0.31)     1.43(0.33)
Min308Ver                (100)   0.90(0.15) |   0.86(0.14)     0.84(0.14) *   0.83(0.14) *   0.82(0.14) *   0.82(0.14) *   0.85(0.15)
Min326Inv                (100)   1.99(0.29) |   1.96(0.28)     1.92(0.28)     1.90(0.28)     1.91(0.27)     1.91(0.27)     2.00(0.27)
Min326Ver                (100)   0.44(0.06) |   0.43(0.06) *   0.43(0.06)     0.43(0.06)     0.43(0.06)     0.43(0.06)     0.44(0.06)
Min338Inv                (100)   1.76(0.25) |   1.74(0.25)     1.74(0.26)     1.77(0.24)     1.78(0.23)     1.80(0.24)     1.94(0.23) v
Min338Ver                (100)   0.97(0.14) |   0.96(0.14)     0.94(0.14)     0.94(0.14) *   0.95(0.14)     0.95(0.14)     0.99(0.14)
Min343Inv                (100)   1.47(0.24) |   1.48(0.24)     1.48(0.24)     1.46(0.24)     1.46(0.24)     1.46(0.24)     1.55(0.23)
Min343Ver                (100)   0.92(0.18) |   0.89(0.17)     0.87(0.17) *   0.87(0.17)     0.86(0.17)     0.86(0.17)     0.85(0.18)
Min348Inv                (100)   1.24(0.16) |   1.24(0.17)     1.26(0.17)     1.26(0.17)     1.26(0.18)     1.25(0.18)     1.31(0.20)
Min348Ver                (100)   0.94(0.20) |   0.92(0.20)     0.92(0.19)     0.91(0.19)     0.90(0.19)     0.89(0.19)     0.88(0.18)
-------------------------------------------------------------------------------------------------------------------------------------------
                               (v/ /*) |         (0/9/1)        (0/8/2)        (0/8/2)        (0/9/1)        (0/9/1)        (1/9/0)


Key:
(1) lazy.IBk '-K 3 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(2) lazy.IBk '-K 4 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(3) lazy.IBk '-K 5 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(4) lazy.IBk '-K 6 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(5) lazy.IBk '-K 8 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(6) lazy.IBk '-K 9 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -30801
(7) lazy.IBk '-K 18 -W 0 -I -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -3080
```

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 4:19 AM


Dataset                   (1) lazy.IBk '-K  | (2) lazy.IBk ' (3) lazy.IBk ' (4) lazy.IBk ' (5) lazy.IBk ' (6) lazy.IBk ' (7) lazy.IBk
-------------------------------------------------------------------------------------------------------------------------------------
Min308Inv                 (100)   1.34(0.30) |   1.34(0.29)     1.36(0.29)     1.37(0.30)     1.37(0.30)     1.37(0.32)     1.47(0.34)
Min308Ver                 (100)   0.90(0.16) |   0.87(0.14)     0.85(0.14)     0.84(0.14) *   0.83(0.14) *   0.83(0.13) *   0.88(0.15)
Min326Inv                 (100)   2.00(0.29) |   1.98(0.28)     1.95(0.29)     1.94(0.28)     1.95(0.27)     1.96(0.28)     2.06(0.27)
Min326Ver                 (100)   0.45(0.06) |   0.43(0.06)     0.44(0.06)     0.44(0.06)     0.44(0.06)     0.44(0.06)     0.45(0.06)
Min338Inv                 (100)   1.78(0.25) |   1.76(0.26)     1.76(0.27)     1.81(0.25)     1.83(0.24)     1.85(0.24)     2.00(0.24)
Min338Ver                 (100)   0.99(0.15) |   0.98(0.14)     0.97(0.15)     0.95(0.14)     0.97(0.14)     0.98(0.14)     1.02(0.14)
Min343Inv                 (100)   1.49(0.25) |   1.52(0.25)     1.52(0.24)     1.50(0.25)     1.50(0.25)     1.50(0.24)     1.62(0.23)
Min343Ver                 (100)   0.93(0.18) |   0.90(0.17) *   0.88(0.17) *   0.88(0.17)     0.87(0.17)     0.87(0.17)     0.87(0.17)
Min348Inv                 (100)   1.26(0.17) |   1.28(0.18)     1.30(0.18)     1.30(0.18)     1.30(0.18)     1.30(0.18)     1.36(0.20)
Min348Ver                 (100)   0.95(0.20) |   0.94(0.20)     0.94(0.19)     0.93(0.19)     0.92(0.19)     0.91(0.19)     0.91(0.18)
-------------------------------------------------------------------------------------------------------------------------------------
                              (v/ /*) |        (0/9/1)        (0/9/1)        (0/9/1)        (0/9/1)        (0/9/1)        (3/7/0


Key:
(1) lazy.IBk '-K 3 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(2) lazy.IBk '-K 4 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(3) lazy.IBk '-K 5 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(4) lazy.IBk '-K 6 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(5) lazy.IBk '-K 8 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(6) lazy.IBk '-K 9 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
(7) lazy.IBk '-K 18 -W 0 -F -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.ManhattanDistance -R first-last\\\"\"' -308
```

```
Analysing:  Mean_absolute_error
Datasets:   10
Resultsets: 9
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       6/18/14 6:38 PM


Dataset                 (1) functions.Mul | (2) functions. (3) functions. (4) functions. (5) functions. (6) functions. (7) functions. (8) functions. (9) functions.
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
Min308Inv               (100)   0.89(0.19) |   0.87(0.19)     0.97(0.22)     1.04(0.22) v   1.57(0.34) v   1.43(0.34) v   2.63(0.67) v   1.42(0.37) v   2.63(0.67) v
Min308Ver               (100)   0.74(0.12) |   0.73(0.11)     0.73(0.12)     0.99(0.22) v   1.11(0.30) v   1.11(0.30) v   1.43(0.31) v   1.12(0.30) v   1.43(0.31) v
Min326Inv               (100)   1.43(0.20) |   1.41(0.19)     1.50(0.20)     1.50(0.22)     2.00(0.54) v   1.96(0.36) v   1.79(0.36) v   1.99(0.41) v   3.58(0.67) v
Min326Ver               (100)   0.45(0.07) |   0.44(0.07)     0.46(0.06)     0.46(0.07)     0.57(0.14) v   0.58(0.20) v   0.60(0.16) v   0.58(0.13) v   0.60(0.16) v
Min338Inv               (100)   1.61(0.18) |   1.59(0.16)     1.61(0.18)     1.63(0.20)     2.02(0.38) v   2.04(0.34) v   2.85(0.62) v   2.13(0.38) v   3.04(0.50) v
Min338Ver               (100)   0.97(0.13) |   0.95(0.12)     0.94(0.14)     0.98(0.16)     1.15(0.25) v   1.13(0.25) v   1.11(0.26)     1.10(0.25)     1.94(0.45) v
Min343Inv               (100)   1.06(0.14) |   1.05(0.14)     1.08(0.13)     1.09(0.14)     1.43(0.34) v   1.39(0.27) v   2.00(0.99) v   1.37(0.27) v   2.96(0.49) v
Min343Ver               (100)   0.87(0.24) |   0.84(0.18)     0.84(0.21)     0.85(0.24)     1.01(0.35)     1.00(0.34)     0.97(0.36)     1.02(0.37)     1.80(0.45) v
Min348Inv               (100)   1.07(0.17) |   1.05(0.16)     1.06(0.15)     1.11(0.17)     1.39(0.38) v   1.32(0.27) v   2.00(0.73) v   1.34(0.31) v   2.45(0.45) v
Min348Ver               (100)   0.83(0.19) |   0.81(0.19)     0.86(0.18)     0.84(0.19)     1.04(0.33) v   1.05(0.26) v   1.15(0.33) v   1.10(0.25) v   1.48(0.35) v
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
                                 (v/ /*) |       (0/10/0)       (0/10/0)       (2/8/0)        (9/1/0)        (9/1/0)        (8/2/0)        (8/2/0)        (10/0/0)


Key:
(1) functions.MultilayerPerceptron '-L 0.08 -M 0.04 -N 1500 -V 0 -S 0 -E 20 -H 1' -5990607817048210779
(2) functions.MultilayerPerceptron '-L 0.04 -M 0.1 -N 2500 -V 0 -S 0 -E 20 -H 1' -5990607817048210779
(3) functions.MultilayerPerceptron '-L 0.04 -M 0.1 -N 2500 -V 0 -S 0 -E 20 -H 4' -5990607817048210779
(4) functions.MultilayerPerceptron '-L 0.08 -M 0.04 -N 1500 -V 0 -S 0 -E 20 -H \"7, 1\"' -5990607817048210779
(5) functions.MultilayerPerceptron '-L 0.4 -M 0.2 -N 1500 -V 0 -S 0 -E 20 -H \"7, 1\"' -5990607817048210779
(6) functions.MultilayerPerceptron '-L 0.4 -M 0.2 -N 1500 -V 0 -S 0 -E 20 -H \"7, 7, 7\"' -5990607817048210779
(7) functions.MultilayerPerceptron '-L 0.4 -M 0.2 -N 1500 -V 0 -S 0 -E 20 -H \"7, 7, 7, 7\"' -5990607817048210779
(8) functions.MultilayerPerceptron '-L 0.4 -M 0.2 -N 1500 -V 0 -S 0 -E 20 -H \"7, 7\"' -5990607817048210779
(9) functions.MultilayerPerceptron '-L 0.4 -M 0.2 -N 1500 -V 0 -S 0 -E 20 -H \"7, 7, 1, 1\"' -5990607817048210779
```

Anexo 2: Archivo que muestra los resultados de pronóstico por el modelo existente en el CMPVC para el período Marzo 2011 para todas las estaciones en Cuba. Las estaciones para la provincia de Villa Clara están señaladas.

	Estacion		Total Casos	Total Casos Positivos	% Casos Positivos	Error Medio	Error Absoluto Medio
2	78361	30	27	90	0.355817633333334	1.0394005	
3	78360	31	28	90.3225806451613	0.331731580645161	0.842909451612903	
4	78363	31	25	80.6451612903226	-0.923342612903226	1.54151925806452	
5	78362	29	29	100	0.200622379310345	0.752314655172414	
6	78365	31	28	90.3225806451613	-0.474560580645161	0.949362580645161	
7	78364	31	30	96.7741935483871	0.0524604838709676	0.675174935483871	
8	78366	31	28	90.3225806451613	0.183491322580645	0.996317387096774	
9	78339	31	27	87.0967741935484	-0.200215548387097	0.97360070967742	
10	78378	31	29	93.5483870967742	-0.317241225806452	0.809517419354839	
11	78331	31	29	93.5483870967742	0.288253129032258	1.04270312903226	
12	78330	31	30	96.7741935483871	-0.267319419354839	0.838765290322581	
13	78333	31	27	87.0967741935484	-0.166453903225807	1.06134674193548	
14	78326	31	24	77.4193548387097	-0.251827225806452	1.38805451612903	
15	78316	31	24	77.4193548387097	-0.400707483870968	1.19929780645161	
16	78338	31	17	54.8387096774194	0.776630774193548	2.22155432258064	
17	78376	31	21	67.741935483871	-0.487079064516129	1.81529190322581	
18	78349	31	27	87.0967741935484	0.620318354838709	1.21970977419355	
19	78348	31	24	77.4193548387097	1.10655961290323	1.34960206451613	
20	78308	23	20	86.9565217391304	0.750068086956522	1.26650391304348	
21	78341	31	30	96.7741935483871	0.150631806451613	0.713638838709677	
22	78340	31	18	58.0645161290323	0.204116516129032	1.83858193548387	
23	78343	31	26	83.8709677419355	0.400983387096774	0.968560290322581	
24	78342	31	27	87.0967741935484	0.658437935483871	0.864524774193548	
25	78345	31	27	87.0967741935484	0.238043548387097	1.00839606451613	
26	78321	31	16	51.6129032258064	1.52012877419355	1.72354096774194	
27	78320	31	25	80.6451612903226	-0.272153870967742	1.33777283870968	
28	78323	31	22	70.9677419354839	-0.418709451612903	1.43739422580645	
29	78322	31	20	64.5161290322581	0.00986403225806424	1.60415487096774	
30	78325	31	30	96.7741935483871	-0.00420683870967684	0.948939935483871	
31	78373	31	18	58.0645161290323	0.0895487419354839	1.80079209677419	
32	78327	31	25	80.6451612903226	-0.703033193548387	1.46085125806452	
33	78335	31	21	67.741935483871	-1.07454841935484	1.43633667741935	
34	78334	31	17	54.8387096774194	-0.721833193548387	1.77384732258064	

Anexo 3: Resultados obtenidos por la aplicación SAGI para las cincos estaciones que se trata en este trabajo.

```
esto es path----C:\Users\Desktop\MisDatos\Marzo\Marzo308Inv.arff
REAL            SAGI        EMA
17.5            18.5        -1.0
18.3            18.8        -0.5
16.9            16.8        0.1
17.4            17.4        -0.0
18.8            17.9        0.9
19.0            18.5        0.5
16.1            17.3        -1.2
16.0            14.9        1.1
16.1            15.5        0.6
17.0            16.6        0.4
17.5            17.2        0.3
16.7            16.2        0.5
15.8            15.8        -0.0
17.8            17.3        0.5
16.5            16.1        0.4
15.3            14.6        0.7
16.5            15.9        0.6
16.6            16.5        0.1
18.0            18.5        -0.5
21.5            19.6        1.9
19.5            19.1        0.4
18.3            19.7        -1.4
19.1            19.4        -0.3
Error Medio Absoluto 0.664164270225191
================================================
Total number of cases is:   23.0
================================================
Total number of positive2 cases is: 23.0
Percentage of Positive2 Cases is :  100.0 Percentage
================================================
Total number of positive3 cases is: 23.0
Percentage of Positive3 Cases is :  100.0 Percentage
================================================
================================================
```

```
esto es path----C:\Users\Desktop\MisDatos\Marzo\Marzo2011_338Inv.arff
REAL            SAGI        EMA
17.0            16.5         0.5
20.4            20.4        -0.0
19.4            19.0         0.4
19.0            18.8         0.2
18.9            16.3         2.6
16.4            14.8         1.6
17.3            15.5         1.8
18.5            15.6         2.9
20.7            20.1         0.6
15.0            15.9        -0.9
14.7            15.1        -0.4
9.0             10.1        -1.1
19.0            17.8         1.2
19.4            17.1         2.3
13.4            14.8        -1.4
15.3            15.5        -0.2
18.7            15.6         3.1
20.2            20.1         0.1
13.8            15.0        -1.2
16.7            16.1         0.6
20.5            19.2         1.3
16.2            16.3        -0.1
13.0            12.9         0.1
14.2            14.1         0.1
12.5            12.9        -0.4
14.0            14.7        -0.7
17.0            17.8        -0.8
18.3            18.2         0.1
21.5            20.6         0.9
20.0            20.0        -0.0
19.8            19.8         0.0
Error Medio Absoluto 1.8348512762259364
================================================
Total number of cases is:   31.0
================================================
Total number of positive2 cases is: 27.0
```

```
esto es path----C:\Users\Desktop\MisDatos\Marzo\Marzo343.arff
REAL            SAGI        EMA
17.0            14.9        2.1
19.5            18.9        0.6
17.0            16.1        0.9
15.8            16.6        -0.8
17.5            16.9        0.6
15.5            15.3        0.2
16.3            16.0        0.3
16.2            15.1        1.1
18.1            17.7        0.4
15.0            16.7        -1.7
11.1            12.2        -1.1
15.4            13.8        1.6
15.2            14.5        0.7
14.5            15.6        -1.1
14.7            17.0        -2.3
15.1            16.7        -1.6
16.1            16.9        -0.8
15.5            15.5        0.0
14.2            15.2        -1.0
16.0            16.7        -0.7
15.7            16.8        -1.1
15.0            15.5        -0.5
16.3            16.3        0.0
14.3            14.8        -0.5
16.7            16.2        0.5
18.4            18.4        -0.0
19.8            19.4        0.4
16.6            16.7        -0.1
20.0            19.1        0.9
16.1            16.7        -0.6
15.9            16.8        -0.9
Error Medio Absoluto 0.9169166795796209
=================================================
Total number of cases is:   31.0
=================================================
Total number of positive2 cases is: 29.0
```

```
esto es path----C:\\Desktop\MisDatos\Marzo\March326.arff
REAL            SAGI        EMA
17.1            14.3        2.8
18.9            17.9        1.0
16.6            13.5        3.1
14.0            14.7        -0.7
16.2            15.6        0.6
15.6            15.9        -0.3
17.9            14.9        3.0
14.8            11.5        3.3
16.2            16.6        -0.4
13.6            16.7        -3.1
6.9             9.4         -2.5
13.8            10.8        3.0
12.5            11.8        0.7
12.0            13.4        -1.4
14.0            16.4        -2.4
13.7            11.0        2.7
14.5            15.4        -0.9
12.1            13.5        -1.4
11.7            13.2        -1.5
15.5            15.3        0.2
15.6            15.6        0.0
12.4            13.4        -1.0
12.8            14.5        -1.7
12.8            13.5        -0.7
12.9            14.5        -1.6
18.5            17.2        1.3
17.9            19.0        -1.1
18.9            17.0        1.9
20.2            18.3        1.9
19.4            20.3        -0.9
20.2            20.3        -0.1
Error Medio Absoluto 1.6621376077237202
================================================
Total number of cases is:   31.0
================================================
Total number of positive2 cases is: 22.0
```

```
esto es path----C:\Users\Desktop\MisDatos\Marzo\Marzo_348Inv.arff
REAL              SAGI        EMA
19.5              19.1         0.4
21.7              21.7         0.0
21.0              20.8         0.2
20.7              20.9        -0.2
19.2              19.5        -0.3
19.5              20.1        -0.6
20.5              21.0        -0.5
19.5              19.5         0.0
19.1              19.0         0.1
17.6              17.8        -0.2
17.6              17.8        -0.2
19.9              19.7         0.2
19.5              19.5         0.0
20.5              20.2         0.3
17.3              18.0        -0.7
18.0              19.5        -1.5
21.5              21.3         0.2
20.8              20.3         0.5
20.6              20.4         0.2
21.6              21.4         0.2
20.6              20.2         0.4
17.6              18.0        -0.4
18.5              18.6        -0.1
16.5              16.5         0.0
18.2              18.5        -0.3
20.2              19.8         0.4
21.4              21.3         0.1
21.6              20.9         0.7
21.1              21.0         0.1
21.2              21.2        -0.0
21.9              21.7         0.2
Error Medio Absoluto 0.9153543893449123
=====================================================
Total number of cases is:   31.0
```