FACULTAD DE INGENIERÍA Y CIENCIAS AGROPECUARIAS

MODELO DE PREDICCIÓN DE LA CALIDAD DEL AIRE A PARTIR DE DATOS METEOROLÓGICOS E INFORMACIÓN DEL TRÁFICO AUTOMOVILÍSTICO

AUTORA

JESSIE MARÍA RAMÍREZ SUÁREZ

AÑO

2018

FACULTAD DE INGENIERÍAS Y CIENCIAS AGROPECUARIAS

MODELO DE PREDICCIÓN DE LA CALIDAD DEL AIRE A PARTIR DE DATOS METEOROLÓGICOS E INFORMACIÓN DEL TRÁFICO AUTOMOVILÍSTICO

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniera en Sistemas de Computación e Informática

Profesor Guía

Ph.D. Yves Philippe Rybarczyk

Autora

Jessie María Ramírez Suárez

Año

2018

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido el trabajo, modelo de predicción de la calidad del aire a partir de datos meteorológicos e información del tráfico automovilístico, a través de reuniones periódicas con la estudiante Jessie María Ramírez Suárez, en el semestre 2018-1, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

_____

Yves Philippe Rybarczyk

Doctor en Informática

CI: 1756950976

DECLARACIÓN DEL PROFESOR CORRECTOR

“Declaro haber revisado este trabajo, modelo de predicción de la calidad del aire a partir de datos meteorológicos e información del tráfico automovilístico, de Jessie María Ramírez Suárez, en el semestre 2018-1, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

_____

Bernarda Cecibel Sandoval Romo

Máster en Ciencias de la Computación

CI: 1709974453

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

“Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

_____

Jessie María Ramírez Suárez

CI: 1717484909

AGRADECIMIENTOS

A mi tutor Yves Rybarczyk y al profesor Mario Gonzalez por su ayuda y explicaciones en el transcurso del desarrollo de esta tesis.

DEDICATORIA

A mi familia por ser mi apoyo incondicional, especialmente a mi madre siempre con su esfuerzo y amor para que pueda culminar mis estudios. A mi padre que es muy especial en mi vida.

RESUMEN

La contaminación del aire representa un importante riesgo medioambiental para la salud. Únicamente buscando disminuir los niveles de esta contaminación, los países pueden reducir la carga de morbilidad derivada de accidentes cerebrovasculares, cánceres de pulmón y neumopatías crónicas y agudas, entre ellas el asma. (Organización Mundial de la Salud, 2016). En el artículo “Modeling PM2.5 Urban Pollution Using Machine Learning and Selected Meteorological Parameters” publicado el 18 de Junio de 2017 por Yves Rybarczyk, Mario Gonzalez y Rasa Zalakeiciute, docentes de la Universidad de las Américas; proponen un enfoque de aprendizaje automático basado en seis años de análisis de datos meteorológicos y de contaminación con el fin de predecir las concentraciones de material particulado fino (PM2.5) a partir de los niveles de viento (velocidad y dirección) y precipitación aplicando el algoritmo de Regresión Lineal.

En el presente documento de tesis, se definen conceptos de Machine Learning como: Aprendizaje Supervisado, Clasificadores, Regresión, entre otros temas.

Para obtener la información, se ejecutaron dos aplicaciones en la recolección de datos, los cuales se obtuvieron por los métodos que a continuación se detallan: datos del tiempo de tráfico administrado por Google Maps. Pantallas capturadas de Google Maps, obteniendo información del tráfico representado en colores rojo, naranja y verde, graficados en forma rectangular. Y, por último, se recolectaron datos realizando un corte de área circular. Toda la información obtenida en estos tres métodos se almacenó en un archivo CSV, el mismo que se refresca cada 10 minutos. Ya obtenidos los archivos CSV, se realizó el preprocesamiento de datos procediendo a una limpieza de la información llenando los espacios vacíos de los archivos CSV con el signo **“?”** para cargarlos en el software Weka, se trabajó con Regresión Lineal, los clasificadores como Redes Neuronales, SVM, Regresión Logística y KNN. Con los resultados obtenidos se elaboraron matrices de confusión para la construcción de la curva ROC a fin de obtener la técnica de aprendizaje supervisado con mejor desempeño.

A futuro se podrá desarrollar una aplicación, con el objetivo de que este ente conozca el nivel de contaminación al que la población se expone y tomar medidas correctivas.

ABSTRACT

Air pollution poses a major environmental risk to health. Only by seeking to reduce the levels of this pollution, countries can reduce the burden of disease resulting from stroke, lung cancers and chronic and acute pneumopathies, including asthma. (World Health Organization, 2016). In the article "Modeling PM2.5 Urban Pollution Using Machine Learning and Selected Meteorological Parameters "published on June 18,2017 by Yves Rybarczyk, Mario Gonzalez and Rasa Zalakeiciute, professors at the University of the Americas; they propose an automatic learning approach based on six years of analysis of meteorological data and pollution in order to predict concentrations of fine particulate matter (PM2).5) from wind levels (speed and direction) and precipitation by applying the Linear Regression algorithm.

In this thesis document, Machine Learning concepts are defined as: Supervised Learning, Classifiers, Regression, among other topics.

In order to obtain the information, two applications were executed in data collection, which were obtained by the following methods: traffic time data managed by Google Maps. Screens captured from Google Maps, obtaining traffic information represented in red, orange and green, graphically rectangular shape. And finally, data was collected by making a circular area cut. All the information obtained in these three methods was stored in a CSV file, which is refreshed every 10 minutes. Once the CSV files had been obtained, the data preprocessing was carried out, cleaning the information by filling the empty spaces of the CSV files with the "?" sign. to load them into the Weka software, we worked with Linear Regression, classifiers such as Neural Networks, SVM, Logistic Regression and KNN. Using the results obtained, confusion matrices were developed for the construction of the ROC curve in order to obtain the best performing supervised learning technique.

In the future, an application can be developed with the aim of making this entity aware of the level of contamination to which the population is exposed and taking corrective measures.

ÍNDICE

1. Capítulo I. Introducción

1.1 Antecedentes

La contaminación es la incorporación al medio ambiente de agentes nocivos en cualquier estado y de origen tanto biológico, como físico y químicos peligrosos para la salud de los seres humanos, animales y plantas. (phs Serkoten, 2017).

Sin embargo, el tráfico automovilístico es un problema predominante, principalmente en zonas urbanas. (OMS, 2014).

De acuerdo con la Organización Mundial de la Salud de 7,4 mil millones de personas (duplicado desde 1970) se espera que cruce un umbral de 9,7 mil millones en los próximos 35 años. Los efectos del rápido crecimiento de la población se reflejan en el uso excesivo y la escasez de recursos naturales, la deforestación, el cambio climático y contaminación ambiental.

Según la última base de datos sobre calidad del aire urbano, el 98% de las ciudades de países de ingresos bajos y medios con más de 100.000 habitantes no cumplen con los requisitos de la Organización Mundial de la Salud (OMS). Un estudio reciente sobre la calidad del aire en Quito, la capital del Ecuador coincide en que los niveles a largo plazo de contaminación son significativamente más altos que los estándares nacionales. (Rybarczyk, Zalakeviciute, 2016)

En el artículo “Modeling PM2.5 Urban Pollution Using Machine Learning and Selected Meteorological Parameters” publicado el 18 de Junio de 2017 por Yves Rybarczyk, Mario Gonzalez y Rasa Zalakeiciute, docentes de la Universidad de las Américas; proponen un enfoque de aprendizaje automático basado en seis años de análisis de datos meteorológicos, químicos tales como Rodio (RH), óxidos de nitrógeno expresados como dióxido de nitrógeno (NO2), dióxido de azufre (SO2), monóxido de carbono (CO) y oxidantes fotoquímicos expresados como ozono (O3); y de contaminación con el fin de predecir las concentraciones de material particulado fino (PM2.5) a partir de los niveles de viento (velocidad y dirección) y precipitación aplicando el algoritmo de Regresión Lineal.

Como resultado de la investigación indicada en el artículo, la exactitud obtenida fue de 44.4%.

El presente trabajo pretende extender la investigación mencionada adicionar una nueva variable el tráfico; y aplicar otras técnicas de aprendizaje automático, proyectando con este estudio el optimizar la aceptación referente a la propuesta de un mejoramiento en el rendimiento de los niveles de confianza.

1.2 Alcance

Para poder construir el modelo de predicción, se comenzará desarrollando una aplicación de escritorio que obtendrá y almacenará los datos del tráfico en tiempo real de Google Maps. Los datos almacenados serán la fecha y hora de la consulta, flujo vehicular en los dos sentidos de cada calle y avenida; y se almacenarán en un archivo CSV, mismo que se irá refrescando cada 10 minutos. Para tomar estos datos se especificarán las coordenadas de origen y destino. En el presente estudio se considerará una de las zonas de la ciudad de Quito con mayor tráfico; el sector del colegio San Gabriel.

A fin de obtener una cantidad de datos razonable que permitan probar el modelo, es necesario realizar como mínimo el almacenamiento de la información por un mes; y debido a su volumen, se alojará en un servidor Linux. Esta información almacenada, será analizada utilizando técnicas de aprendizaje supervisado.

El proceso a realizar será primeramente seleccionar el conjunto de datos, es decir, las variables a predecir, en este caso el tráfico vehicular obtenido de Google Maps en conjunto con los datos meteorológicos, precipitación, radiación solar, presión, temperatura, además de los químicos, CO, RH, NO2, O3 y SO2 que ya fueron analizados en el estudio mencionado. Lo siguiente será la transformación o preprocesamiento del conjunto de datos, en este paso los datos son normalizados, y se decide cómo se van a tratar los faltantes. Posteriormente se aplicarán técnicas de aprendizaje supervisado tales como Regresión Lineal, Regresión Logística, SVM, entre otros para lograr así construir el modelo de predicción. Este modelo permitirá revelar información de la contaminación en la ciudad de Quito.

El modelo construido, será validado a través de una evaluación del resultado obtenido con los datos meteorológicos y químicos adicionando el tráfico, todos estos datos contra PM2.5 (Es el material particulado respirable de 2,5 micrómetros presente en la atmósfera (Ecologista, 2008)). Serán considerados resultados buenos cuando el coeficiente de correlación es mayor a 0.6, en caso de que el resultado final no cumpla con el parámetro, se repetirá el proceso manteniendo la variable del tráfico constante y modificando las demás variables (datos meteorológicos y químicos).

1.3 Justificación

La contaminación del aire representa un importante riesgo medioambiental para la salud. Únicamente buscando disminuir los niveles de esta contaminación, los países pueden reducir la carga de morbilidad derivada de accidentes cerebrovasculares, cánceres de pulmón y neumopatías crónicas y agudas, entre ellas el asma. (Organización Mundial de la Salud, 2016).

Cabe indicar además que, adicional a la contaminación por el impacto humano y automovilístico, y; el rápido crecimiento de la población, influyen también la contaminación del aire por partículas moduladas por factores meteorológicos.

Cuánto más bajos se presenten los niveles de contaminación del aire, la salud cardiovascular y respiratoria de la población será mejor.

Los habitantes de la ciudad de Quito desconocen la magnitud del nivel de contaminación al que se hallan expuestos en el día a día; sin embargo, gracias a la construcción de este modelo de medición podrían conocer los niveles y tomar medidas preventivas o correctivas necesarias.

El uso de modelos basados en aprendizaje supervisado ayudará a predecir la contaminación en el aire a partir de la información del tráfico, relacionado también con datos meteorológicos.

Se propone un modelo de predicción de la calidad del aire, utilizando algoritmos de aprendizaje supervisado con el objetivo de pronosticar el nivel de partículas finas utilizando los factores de viento, precipitación y tráfico.

1.4 Objetivos

1.4.1 Objetivo general

Proponer un modelo de predicción del nivel de contaminación en la ciudad de Quito, a partir de datos meteorológicos y de actividad humana relacionada al tráfico automovilístico.

1.4.2 Objetivos específicos

- Extraer los datos de tráfico automovilístico, utilizando el aplicativo Google Maps, de una zona específica de la ciudad de Quito, donde se produce mayor grado de tráfico vehicular.
- Analizar técnicas de aprendizaje supervisado, para la creación del modelo.
- Reflejar en los resultados obtenidos el mejoramiento en la predicción de la calidad del aire utilizando los factores (datos meteorológicos y de tráfico).

1.5 Metodología por utilizar

Para la construcción del modelo de la predicción de la calidad del aire a partir de datos meteorológicos e información del tráfico automovilístico se utilizarán los siguientes pasos:

- Elegir la experiencia de entrenamiento, donde se estudiarán los datos meteorológicos: velocidad del viento, dirección del viento, precipitación, radiación solar, presión y temperatura. Adicional se estudia la presencia de los químicos en el aire que son: CO, RH, NO2, O3 y SO con una variable adicional,

el tráfico.

- Se aplicarán técnicas de aprendizaje supervisado: Regresión Lineal, Redes Neuronales, SVM y Regresión Logística.
- Evaluar cuál de las técnicas de aprendizaje supervisado es la más recomendable, aplicando matrices de confusión y curvas ROC

2. Capítulo II: Marco Teórico

2.1 Sistemas de aprendizaje automático

Existen varios investigadores que han dado diferentes conceptos sobre sistemas de aprendizaje automático entre los principales se puede mencionar a los siguientes:

Para Statistical Analysis System (SAS), el aprendizaje automático nace gracias a la hipótesis de que las computadoras podrían aprender sin programarse para realizar ciertas tareas. Los interesados en Inteligencia Artificial (IA) quisieron probar si las computadoras, eran capaces de aprender en base a datos.
En este proceso del aprendizaje de máquina, el aspecto reiterativo es importante ya que los modelos son expuestos a la entrada de nuevos datos. Son capaces de aprender de cálculos realizados anteriormente para la toma de decisiones, resultados verídicos y repetibles. (SAS Inc, 2016)

Para García, los sistemas de aprendizaje automático son sistemas informáticos con la capacidad de aprender según la entrada y salida de datos; estos sistemas aprenden mediante la experiencia; es decir, el sistema podrá realizar procesos basados en su comportamiento a una serie de sucesos; y, posteriormente ser capaz de responder en acciones tras sucesos nuevos o imprevistos. (García, 2014).

El sistema de aprendizaje automático se enfoca principalmente en el uso de Redes Neuronales, aprendizaje en base a casos y aprendizaje analítico. Con el paso del tiempo los modelos están siendo utilizados de manera híbrida, simplificando los límites entre ellos y así permitir el desarrollo de modelos

mayormente eficaces. La combinación de métodos analíticos puede garantizar resultados efectivos, reiterativos e íntegros. (García, 2014)

El aprendizaje de máquina se basa en un principio sencillo:

APRENDIZAJE = REPRESENTACIÓN + EVALUACIÓN + OPTIMIZACIÓN

Dónde:

Representación: utiliza un clasificador representado en un lenguaje formal que una computadora puede manejar e interpretar.
Evaluación: consiste en una función necesaria para distinguir o evaluar los clasificadores buenos y malos.
Optimización: representa el método utilizado para buscar entre los clasificadores dentro del lenguaje para encontrar los más altos desempeños. (García, 2014)

2.1.1. Aprendizaje supervisado

El Aprendizaje Supervisado es una estrategia supervisada para identificar las entradas de datos y compararlos con los resultados deseados. (García, 2014)

Se propone el siguiente ejemplo para entender cómo funciona el Aprendizaje Supervisado, se quiere clasificar letras manuscritas, el procedimiento es el siguiente:

1. Se tiene un conjunto de datos que contiene ejemplos de entrenamiento con las etiquetas correctas asociadas. Por ejemplo, cuando se aprende a clasificar dígitos manuscritos, un algoritmo de aprendizaje supervisado toma miles de imágenes de dígitos manuscritos junto con etiquetas que contienen el número correcto que cada imagen representa.

2. El algoritmo entonces aprenderá la relación entre las imágenes y sus números asociados, y aplicará esa relación aprendida para clasificar imágenes

completamente nuevas (sin etiquetas) que la máquina no ha visto antes. (Maini, 2017).

El aprendizaje supervisado se suele aplicar en problemas de clasificación, como identificación de dígitos, diagnósticos, o detección de fraude de identidad. También se usa en problemas de regresión, como predicciones meteorológicas, de expectativa de vida, de crecimiento etc. (Maini, 2017)

2.1.1.1. Algoritmo de clasificación supervisado

La clasificación es cuando el sistema de aprendizaje trata de etiquetar (clasificar) un conjunto de datos para predecir una categoría (clase). La base de conocimiento del sistema está formada por ejemplos de etiquetados anteriores. Este tipo de aprendizaje puede llegar a ser muy útil en problemas de investigación biológica, biología computacional y bioinformática. (Ericson, 2017)

Algunas técnicas de clasificación son:

- KNN (Vecino cercanos)

Los KNN son técnicas de clasificación supervisada, este algoritmo se basa en que el nuevo objeto será clasificado en la clase más usual de sus K vecinos más próximos. (Cárdenas, 2016)

KNN es uno de los algoritmos de clasificación más simples y de los algoritmos de aprendizaje más utilizado. Después de seleccionar el valor de k, puede hacer predicciones basadas en los ejemplos de KNN. En regresión, las predicciones para KNN son el promedio del resultado de los vecinos cercanos a k. (Statsoft, s.f.)

$$y = \frac{1}{K}\sum_{j=1}^{k} y_i \qquad \text{(Ecuación 1)}$$

Donde :

i: muestras, y: es la predicción

- Regresión logística

La Regresión Logística es una técnica multivariable donde la variable dependiente es categórica (verdadero o falso) y las variables independientes son cuantitativas.

Esta técnica es similar al modelo de regresión lineal pero válida para modelos en los que la variable dependiente tiene dos opciones (verdadero o falso).

La variable de salida binaria Y, se modela el condicional de probabilidad Pr (Y = 1|X = x) como función de x; cualquier parámetro desconocido en la función será estimada como probabilidad máxima. (Stat, 2016).

- Redes neuronales

Una red neuronal es un algoritmo de aprendizaje automático basado en el modelo de una neurona humana. Es capaz de aprender a reconocer patrones. Se presentan como sistemas de "neuronas" interconectadas que pueden calcular valores a partir de entradas. Consta de nodos en la analogía biológica representan neuronas, conectados por arcos. Cada arco se relaciona a un peso en cada nodo. Se puede aplicar la red neuronal no sólo para la clasificación, también puede ser aplicado para la regresión de los atributos de objetivo continuo. (Sharma, 2017)

Una red neuronal puede contener las siguientes 3 capas:

Capa de entrada: recibe todos los datos

Capa oculta: se encargan del aprendizaje del algoritmo

Capa de salida: recibe la información de las capas ocultas y entrega un resultado.

Las capas de la red neuronal artificial se representan en la Figura 1.

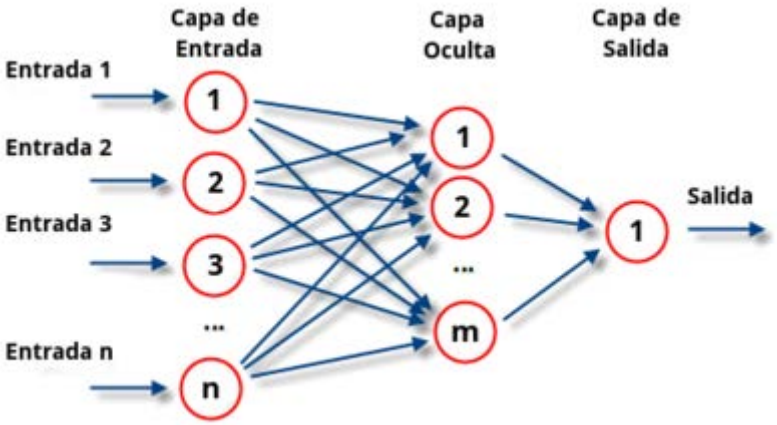

Figura 1. Representación de las capas de una red neuronal artificial.

Aprendizaje de una Red Neuronal:

Se reciben los datos, los nodos de la capa oculta envían la información a las capas de salida, la capa de salida arroja un resultado. Los resultados se analizan contra lo que se estaba esperando, si presenta errores se realizan modificaciones. Cada conexión tiene “x” el dato que envía la capa anterior y “w” es el peso. Bias, es un número que ayuda que ciertas neuronas se activen con mayor facilidad. La neurona sabe que tiene que enviar la información de la siguiente forma, cada neurona suma todos los pesos (w) multiplicado por x, se suma el bias, el resultado es enviado a una función de activación, si el resultado es mayor que cierto número se enviará la información a la siguiente neurona caso contrario no. (Sharma, 2017).

- SVM (Máquinas de vectores de soporte)

Los SVMs, brindan exactitud en la clasificación, se encargan de construir un hiperplano para la separación entre clases lo cual permitirá una correcta clasificación.

La distancia entre el hiperplano y el punto de datos más cercano de cualquiera de los dos conjuntos se conoce como margen. El objetivo es elegir un hiperplano con el mayor margen posible entre el hiperplano y cualquier punto del conjunto de entrenamiento, dando una mayor posibilidad de que los nuevos datos se clasifiquen correctamente. (Kowalczyk, 2014)

2.1.1.2. Regresión Lineal

La regresión lineal es un algoritmo para predecir una respuesta cuantitativa. (Gareth, Witten, Hastie, Tibshirani, 2009, p. 59)

La idea es expresar la clase como una combinación lineal de atributos, con pesos predeterminados:

$$x = w_0 + w_1 a_1 + w_2 a_2 + \cdots + w_k a_k$$

(Ecuación 2)

Donde:

- x es la clase
- $a_1, a_2, \ldots, a_k$ son los valores de atributo
- $w_0, w_1, \ldots, w_k$ son los pesos

El resultado obtenido es un conjunto de pesos numéricos, basado en los datos de entrenamiento, que pueden ser utilizados para predecir la clase de nuevas instancias. (Witthen, Frank, Hall, 2011, p 124-125)

2.1.1.3. Matriz de confusión

Es la técnica para resumir el rendimiento de un algoritmo de clasificación. Al calcular la matriz de confusión se puede tener una mejor idea de lo que el modelo está haciendo correctamente e incorrectamente.

La exactitud de la clasificación por sí sola puede ser engañosa si se tiene un número desigual de observaciones en cada clase o si tiene más de dos clases en un conjunto de datos.

Una matriz de confusión se calcula de la siguiente manera:

1. Se cuenta con los datos reales y predictivos del modelo que se está estudiando en un archivo.

2. Obtener los datos en binario, según los valores reales y predictivos se obtiene una mediana de cada uno, cuando el valor real es mayor a la mediana será 1 cuando es menor será 0 igualmente con el valor predictivo.
3. La matriz de confusión se divide en valores reales y valores predictivos.
4. Clasificar los valores en los diferentes cuadrantes según los siguientes datos:

a. 0-0 Verdaderos Negativos(VN)
b. 0-1 Falsos Negativos (FN)
c. 1-0 Falsos Positivos (FP)
d. 1-1 Verdaderos Positivos (VP).

		Real	
		< 11,96	≥ 11,96
Predictivo	< 11,96	VN	FN
	≥ 11,96	FP	VP

Figura 2. Matriz de confusión.

2.1.1.4. Curva ROC

Las curvas ROC son útiles para comparar diferentes clasificadores. El redimiento de un clasificador es representado por el área bajo la curva (AUC). La curva ROC ideal llega hasta la esquina superior izquierda, por lo cual si el área bajo la curva es más grande es un mejor clasificador. (Gareth, Witten, Hastie, Tibshirani, 2009, p. 147-149)

Se establecen intervalos para los valores de AUC:

- 0.5, 0.6 Test malo
- 0.6, 0.75 Test regular
- 0.75, 0.9 Test bueno
- 0.9, 0.97 Test muy bueno
- 0.97, 1 Test excelente

En la Figura 3 se representan diferentes curvas ROC.

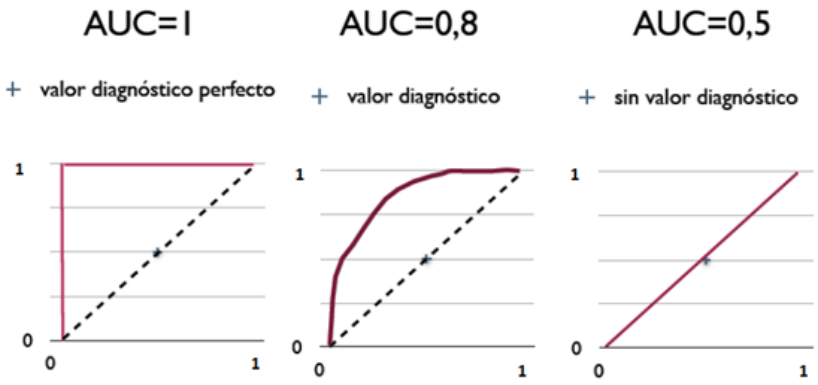

Figura 3. Curvas ROC.
Tomado de Wikipedia, s.f.

Para graficar una curva ROC, se desarrolla una matriz de confusión para obtener los valores de VN, FN, FP y VP como en la Figura 2.

Los cálculos para obtener los valores de la curva ROC son los siguientes:

Sensibilidad o Razón de Verdaderos Positivos (VPR)

$$VPR = \frac{VP}{(VP + FN)} \qquad \text{(Ecuación 3)}$$

Ratio o Razón de Falsos Positivos (FPR)

$$FPR = \frac{FP}{(FP + VN)} \qquad \text{(Ecuación 4)}$$

Especificidad o Razón de Verdaderos Negativos (SPC)

$$SPC = \frac{VN}{(VN + FP)} = 1 - FPR \qquad \text{(Ecuación 5)}$$

2.1.1.5. Herramientas utilizadas para la predicción de la calidad del aire

- Processing

Processing es un lenguaje de programación orientado a diseñadores que no tienen necesariamente que saber programar para usarlo creado por Ben Fry y Casey Reas. Pensado especialmente para proyectos multimedia de diseñadores audiovisuales y como herramienta alternativa al software propietario, ya que se distribuye con licencia GNU GPL. (Rodríguez, 2013)

- Weka

Es una plataforma de software para aprendizaje Automático y minería de datos escrito en Java y desarrollado en la Universidad de Waikato. Weka es un software libre distribuido bajo licencia GNU-GPL. (Ecured, 2016).

3. Capítulo III: Diseño del modelo de predicción de la calidad del aire

Tal como se explicó en el capítulo 1.5, para poder elaborar el modelo de predicción lo primero es la obtención de los datos. En el siguiente orden:

1. Plantear coordenadas de origen y destino dentro de la zona del Colegio San Gabriel.
2. Recolectar datos cada 10 minutos del tiempo del tráfico.
3. Recolectar imágenes del tráfico cada 10 minutos.
4. En el preprocesamiento y exploración de los datos se trataron los faltantes de estos, estos datos fueron reemplazados por el signo “?”.
5. En la selección de atributos se separó el análisis:

a. Datos meteorológicos: radiación solar, temperatura, presión, precipitación, viento.
b. Químicos: CO, RH, NO2, O3, SO2 y tráfico,
c. Datos meteorológicos y químicos, y, por último
d. Datos meteorológicos y tráfico.

6. En el conjunto de entrenamiento y validación, se indica que cross-validation se ejecuta 10 veces.

3.1 Adquisición de datos (ETL)

Para la adquisición de datos se consideraron 2 métodos:

1. Adquisición de tiempos de tráfico

2. Adquisición de imágenes de tráfico

3.1.1. Adquisición de tiempos de tráfico

La representación del tráfico se lo realiza a través de minutos por Google Maps, para la recolección de datos, se ejecutó una aplicación cada 10 minutos programada en Python y almacenada en un servidor Linux; y, se recolectaron datos en los meses de Junio y Julio del año 2017 las 24 horas del día. Se utilizaron las coordenadas en el centro de Belisario en la ciudad de Quito, los datos fueron recogidos del punto 1 (calle Mañosca hacia Av. América), el punto 2 (Av. América hacia Av. Mariana de Jesús), el punto 3 (Av. Mariana de Jesús hacia Av. Antonio José de Sucre), el punto 4 (Av. Antonio José de Sucre hacia Calle Mañosca) como se observa en la Figura 4. Los datos obtenidos se almacenaron en un archivo CSV, en las columnas Date-Time(día y hora de la consulta), N-WE time(tiempo de ida en la calle Mañosca), N-EW time(tiempo de regreso en la calle Mañosca), W-NS time (tiempo de ida en la Avenida Antonio José de Sucre), W-SN time (tiempo de regreso en la Avenida Antonio José de Sucre), S-WE time (tiempo de ida en la Avenida Mariana de Jesús), S-EW time (tiempo de regreso en la Avenida Mariana de Jesús), E-NS time (tiempo de ida en la Avenida América), E-SN time (tiempo de regreso en la Avenida América). En el Anexo 1 se presenta el código completo para la extracción de datos.

Figura 4. Cuadrante donde se recopilaron los datos.

3.1.2. Adquisición de imágenes de tráfico

El tráfico también es representado en colores, rojo, naranja y verde por Google Maps. El color rojo representa que existen retrasos en el tráfico, el color naranja representa que existe una cantidad media de tráfico y el color verde representa que no existen retrasos en el tráfico. Para obtener las imágenes se ejecutó un programa cada 10 minutos que realiza screenshots del tráfico, el programa fue realizado en Processing. Obtenidas las imágenes, se ejecutó un programa para que realice cortes de área circular de las imágenes mencionadas anteriormente, como se puede apreciar en la Figura 5. Se realizaron cortes circulares para analizar si en un área circular se obtienen mejores resultados que en un área rectangular. Se ejecutó otro programa para extraer los pixeles de cada color, y esta información almacenar con las columnas: Número de Imagen, Fecha y Hora en un archivo CSV. En el Anexo 2 se presenta el código completo para la adquisición de imágenes del tráfico.

Figura 5. Captura de pantalla de área circular.

Se realizaron cinco cortes a la captura de pantalla original para así trabajar con diferentes áreas y observar en qué área se obtienen mejores resultados. Las

áreas mencionadas fueron de 2.2 km^2, 5.0 km^2, 8.7 km^2, 10.9 km^2 y 14.0 km^2. En la Figura 6 se presenta la captura original de 14km^2.

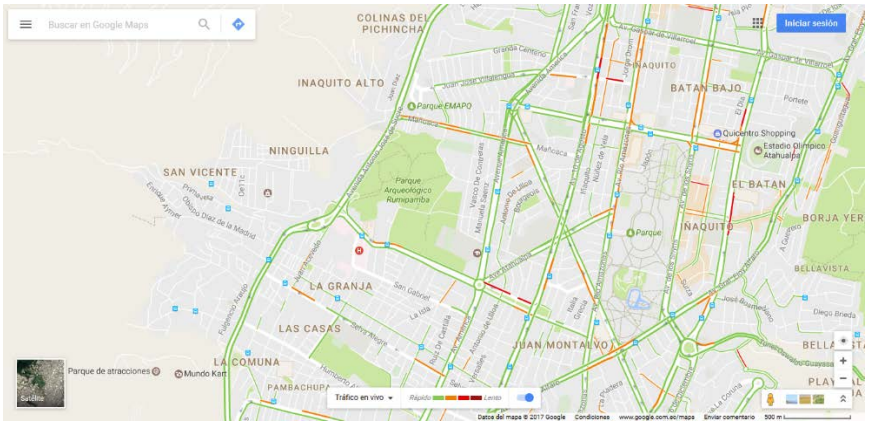

Figura 6. Captura de pantalla original del tráfico representada en colores.

Como se puede observar en la Figura 7; se presenta el primer corte realizado a la captura de pantalla original de área rectangular de 2.2km^2 para el análisis de los pixeles de cada color representados en cada imagen.

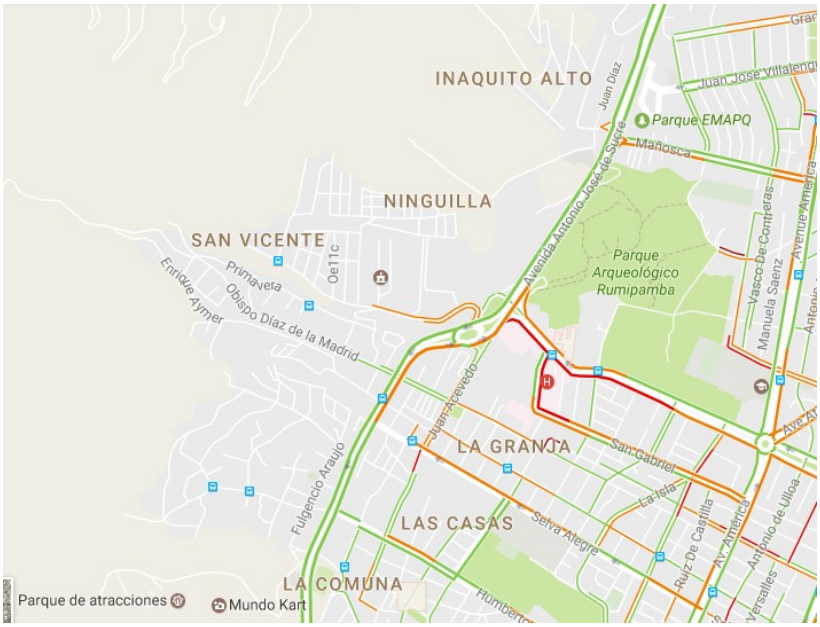

Figura 7. Captura de pantalla de área rectangular de 2.2km^2.

En la Figura 8, se presenta el segundo corte realizado a la captura de pantalla original de área rectangular de 5.0km^2 para el análisis de los pixeles de cada color representados en cada imagen.

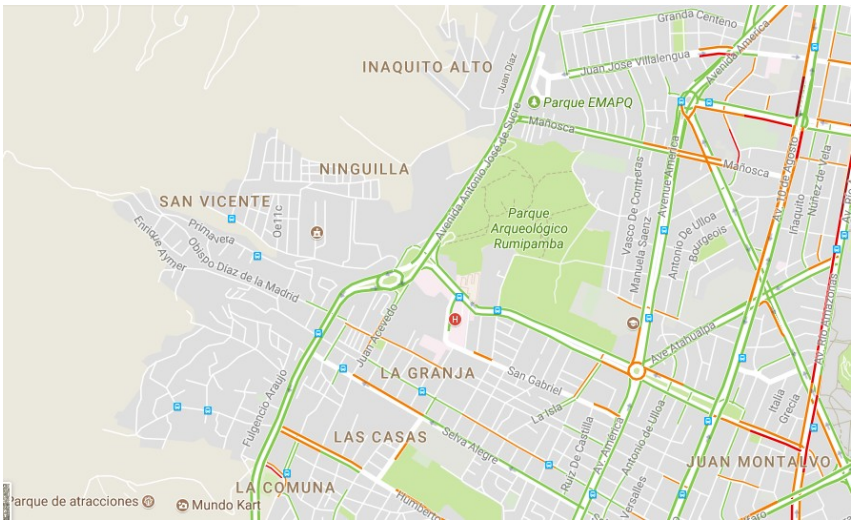

Figura 8. Captura de pantalla de área rectangular de 5.0km^2.

A través de la Figura 9, se presenta el tercer corte realizado a la captura de pantalla original de área rectangular de 8.7km^2 para el análisis de los pixeles de cada color representados en cada imagen.

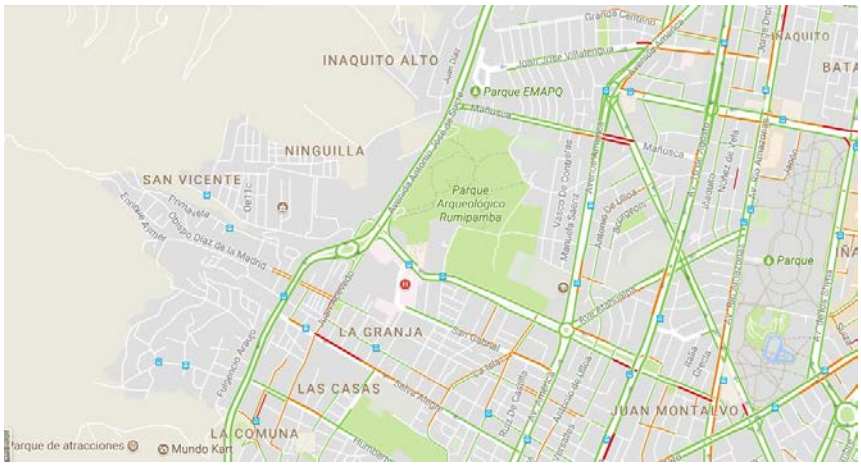

Figura 9. Captura de pantalla de área rectangular de 8.7km^2.

Como se puede observar en la Figura 10, se presenta el cuarto corte realizado a la captura de pantalla original de área rectangular de 10.9km^2 para el análisis de los pixeles de cada color representados en cada imagen.

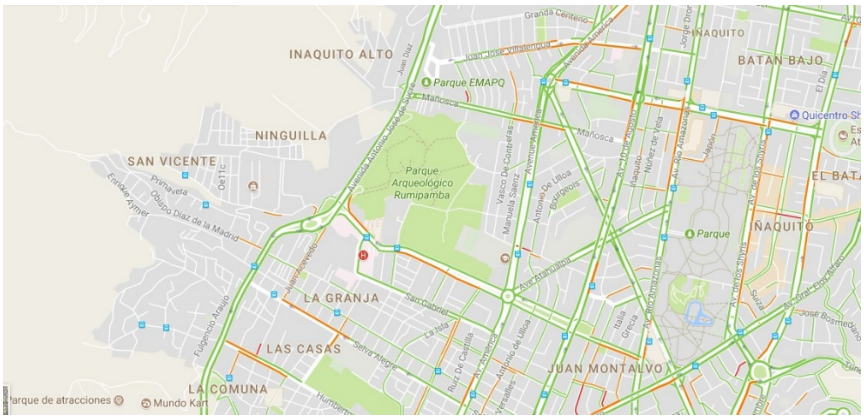

Figura 10. Captura de pantalla de área rectangular de 10.9km^2.

Luego en la Figura 11, se presenta el quinto corte realizado a la captura de pantalla original de área rectangular de 14.0km^2 para el análisis de los pixeles de cada color representados en cada imagen.

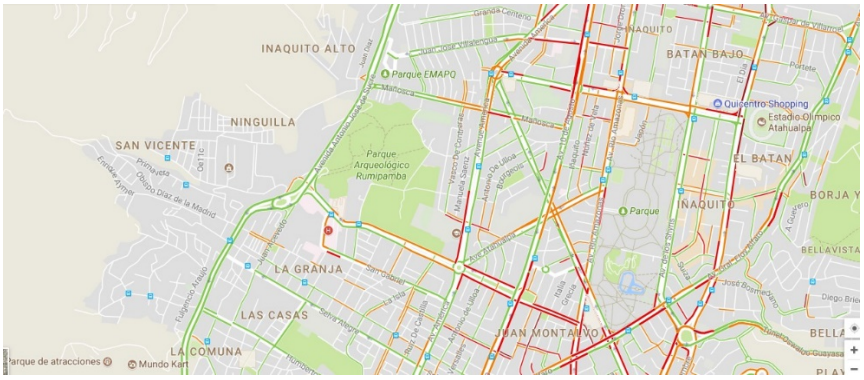

Figura 11. Captura de pantalla de área rectangular de 14.0km^2.

3.2. Preprocesamiento y Análisis Exploratorio de los Datos

Primero se analizaron los tiempos de tráfico adquiridos. Obtenido el archivo CSV en cada hora se almacenaban seis, cinco, cuatro y dos valores de tiempo debido a que el archivo CSV se refrescaba cada 10 minutos y una hora en minutos son 60, seis valores eran almacenados si en toda la hora eran almacenados correctamente, el número de valores bajaba si existía algún contratiempo dentro

de la ejecución de la aplicación para obtener el tiempo de tráfico de Google Maps, uno de los contratiempos fue que Google Maps tomaba a la aplicación como un bot por lo cual la aplicación se reiniciaba en ciertos tiempos. Se calculó el promedio de cada hora para cada calle y avenida obteniendo así las columnas P-N-WE (promedio del tiempo de ida en la calle Mañosca), P-N-EW (promedio del tiempo de regreso en la calle Mañosca), P-W-NS (promedio del tiempo de ida en la Avenida Antonio José de Sucre), P-W-SN (promedio del tiempo de regreso en la Avenida Antonio José de Sucre), P-S-WE (promedio del tiempo de ida en la Avenida Mariana de Jesús), P-S-EW (promedio del tiempo de regreso en la Avenida Mariana de Jesús), P-E-NS (promedio del tiempo de ida en la Avenida América), P-E-SN (promedio del tiempo de regreso en la Avenida América) con los datos meteorológicos Bel_SolarRadiation (Radiación Solar de Belisario), Bel_temperature (Temperatura de Belisario), Bel_atmPressure (Presión Atmosférica de Belisario), Bel_Rain (Precipitación de Belisario), Bel_RelativeHumidity (Humedad de Belisario), Bel_WindSpeed (Velocidad de viento de Belisario), Bel_WindDirection (Dirección del viento de Belisario), Bel_WindX (Viento en x de Belisario), Bel_WindY (Viento en y de Belisario) y PM2.5 (Material Particulado Fino). Los valores de las columnas Bel_WindX y Bel_WindY se obtienen de la siguiente manera:

$$\text{Bel\_WindX} = cos\frac{(Bel\_WindDirection \times \pi)}{180^{\circ}} \times Bel\_WindSpeed \qquad \text{(Ecuación 6)}$$

$$\text{Bel\_WindY} = sin\frac{(Bel\_WindDirection \times \pi)}{180^{\circ}} \times Bel\_WindSpeed \qquad \text{(Ecuación 7)}$$

Los valores que se encuentren en blanco en el archivo CSV deben ser reemplazados por el signo “?”.

Obtenidas las imágenes para identificar la densidad de tráfico, se extraen tres categorías de colores pixelados: verde, naranja y rojo. Los píxeles verde, naranja y rojo significan bajo, medio y alta cantidad de tráfico, respectivamente. El número de píxeles de cada categoría se obtiene por medio del componente RGB de todos los píxeles de la imagen. Después de excluir los píxeles blancos (línea 24, Figura 12), se implementan tres reglas para clasificar los píxeles restantes en una o más de las siguientes opciones otra categoría (líneas 25 a 30 de la

Figura 12). Una vez que la imagen es leída en su totalidad, el porcentaje de cada categoría se calcula dividiendo el número de verde, naranja y rojo píxeles por el número total de píxeles coloreados. En el Anexo 3 se encuentra el código completo de la extracción de pixeles en las capturas de Google Maps.

```
for(int i = 0; i < QFrame; i++) {
  float red = 0;
  float orange = 0;
  float green = 0;
  image(frame[i], 0, 0);
  for(int x=0; x<width; x++) {
    for(int y=0; y<height; y++) {
      color c = get(x, y);
      if(red(c) < 200 || green(c) < 200 || blue(c) < 200) { // remover pixeles blancos
        if(red(c) > green(c) && abs(green(c)-blue(c)) < 20) {
          red++;
        } else if(red(c) > green(c) && green(c) > blue(c)) {
          orange++;
        } else if(green(c) > red(c) && red(c) > blue(c)) {
          green++;
```

Figura 12. Código para extraer información de pixeles de imágenes.

Obtenido el archivo CSV, se tomó el promedio de los porcentajes de cada hora. Las columnas presentadas en el archivo CSV son red, orange, CO, RH, rain, NO2, O3, pressure, SolarRad, SO2, temp, WS, WX, WY y PM2.5.

3.3 Diseño del modelo de aprendizaje

3.3.1. Selección de atributos

Se pueden considerar tres enfoques posibles para predecir el nivel de PM2.5 a partir de los atributos obtenidos:

1. Trabajar con todos los atributos del archivo CSV: red, orange, CO, RH, rain, NO2, O3, pressure, SolarRad, SO2, temp, WS, WX, WY y PM2.5 para todo el día
2. Trabajar con todos los atributos del archivo CSV en la mañana en el horario de 6 am a 10 am que son las horas más transitadas.
3. Trabajar con datos meteorológicos en la mañana en el horario de 6 am a 10 am.

3.3.2. Conjuntos de entrenamiento y validación

Los modelos son entrenados y probados de acuerdo con una validación cruzada de 10 veces.

Cross-validation trabaja de la siguiente manera, el conjunto de datos obtenido es dividido en subconjuntos k. Uno de los subconjuntos k se utiliza como equipo de pruebas y los otros subconjuntos k-1, como un conjunto de entrenamiento. Después, el error promedio es calculado en todos los ensayos k. Cada punto de datos llega a estar en un conjunto de pruebas exactamente una vez, y llega a estar en un conjunto de entrenamiento k-1 veces. (Witten, Frank, Hall, 2011, p.152-153)

4. Capítulo IV: Aplicación del sistema de aprendizaje automático a la predicción de calidad de aire

4.1. Aplicación de algoritmos de aprendizaje supervisado

El rendimiento de los modelos es evaluado por dos métricas: el parámetro coeficiente de correlación, y el error al cuadrado de la media de la raíz. El coeficiente de correlación (r) mide la fuerza de la relación lineal entre dos o más variables. La ventaja de r sobre las otras métricas se basa en una escala con un máximo (± 1) y un mínimo de (0) para cuantificar la fuerza de la relación. Cuanto más cerca de 1 es el valor absoluto, mejor es la correlación. El error de la media cuadrática de la raíz (RMSE) es la raíz cuadrada de la variable error cuadrado promedio por predicción (MSE). RMSE es una métrica de evaluación intuitiva que se utiliza con frecuencia, porque proporciona un rendimiento en la misma unidad que el atributo pronosticado de ella misma. Cuanto menor sea el valor de RMSE, más preciso será el pronóstico del modelo.

4.1.1. Primer método: Tiempo de tráfico + datos meteorológicos + químicos aplicando el algoritmo Regresión Lineal en Weka

4.1.1.1. Tráfico día completo

Uno de los principales objetivos de un enfoque de aprendizaje automático, es producir la información más precisa posible y, que la predicción sea lo más simple posible con un modelo. Se propone construir un modelo basado sólo en el tráfico y evaluando su fiabilidad. El número de atributos son cinco: P-N-WE, P-S-EW, P-E-NS, P-E-SN y PM2.5.

El modelo obtenido en Weka después de ejecutar el algoritmo Regresión Lineal fue:

bel_PM2.5 =

283.8825 * P-N-WE +

-132.4192 * P-S-EW +

-65.0414 * P-E-NS +

104.8096 * P-E-SN +

6.3476

La precisión de predicción del modelo se evalúa como:

r = 0.2625

RMSE = 8.1852

El modelo muestra que el parámetro con mayor peso es P-N-WE. El coeficiente de correlación de este modelo es bastante bajo (r≈0.26).

4.1.1.2. Tráfico en la mañana

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

bel_PM2.5 =

6.5462 * P-N-WE +

-6.595 * P-E-NS +

14.26 * P-E-SN +

-12.6911

La precisión de predicción del modelo se evalúa como:

r = 0.2899

RMSE = 11.5045

El modelo muestra que el parámetro con mayor peso es P-E-SN. El coeficiente de correlación de este modelo es igual de bajo (r≈0.29), se deberá tomar en cuenta que se trabajó con datos en el horario de la mañana de 6 am a 10am.

4.1.1.3. Tráfico y meteorología día completo

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

bel_PM2.5 =

4.7692 * P-N-WE +

-2.5807 * P-S-EW +

4.094 * P-E-SN +

0.4766 * Bel_temperature +

1.6296 * Bel_atmPressure +

-1.3159 * Bel_Rain +

0.1859 * Bel_RelativeHumidity +

-0.9075 * Bel_WindSpeed +

-1201.9693

La precisión de predicción del modelo se evalúa como:

r = 0.4084

RMSE = 11.0579

El modelo muestra que el parámetro con mayor peso es P-N-WE. El coeficiente de correlación de este modelo es un poco más alto comparado con los anteriores (r≈0.4084), tomar en cuenta que se trabajó además con datos meteorológicos.

4.1.1.4. Tráfico y meteorología en la mañana

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

bel_PM2.5 =

6.6304 * P-N-WE +

-5.7086 * P-E-NS +

11.4904 * P-E-SN +

3.6611 * Bel_atmPressure +

-3.8021 * Bel_WindSpeed +

-2668.7159

La precisión de predicción del modelo se evalúa como:

r = 0.4571

RMSE = 7.547

El modelo muestra que el parámetro con mayor peso es P-E-SN. El coeficiente de correlación de este modelo es un poco más alto comparado con los anteriores

(r≈0.4571), considerar que se trabajó con datos meteorológicos y en el horario de mañana de 6 am a 10am.

Tabla 1

Coeficientes de correlación de los modelos obtenidos aplicando el algoritmo de Regresión Lineal.

	r	RMSE	Parámetro con mayor peso
Tráfico día completo	0.2625	8.1852	P-N-WE = 283.8825
Tráfico en la mañana	0.2899	11.5045	P-E-SN = 14.26
Tráfico + datos meteorológicos día completo	0.4084	11.0579	P-N-WE = 4.7692
Tráfico + datos meteorológicos en la mañana	0.4571	7.547	P-E-SN = 11.4904

Conclusión:

En la tabla 1 donde se presentan los coeficientes de correlación de los modelos obtenidos aplicando el algoritmo de Regresión Lineal, se puede ver que el r más alto obtenido fue de 0.4571 trabajando con los datos del tráfico + datos meteorológicos en horas de la mañana. El parámetro con mayor peso en el modelo representa el parámetro que tiene mayor influencia para la obtención de resultados. Al crear un modelo solo con datos del tráfico se obtuvo un r demasiado bajo y al crear el modelo basado en el tráfico + datos meteorológicos se puede indicar que los datos meteorológicos son factores que mejoran los resultados en la predicción de la calidad del aire. También se observa que, al trabajar con la información del tráfico en horas de la mañana se obtienen mejores resultados.

4.1.2. Segundo método: Información de imágenes de tráfico de área circular + datos meteorológicos + químicos aplicando el algoritmo Regresión Lineal en Weka

Ejecución de la aplicación que realiza capturas de pantalla, las imágenes obtenidas fueron recortadas en un área circular. La información de los pixeles de las imágenes es almacenada en un archivo CSV.

4.1.2.1. Tráfico día completo

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

PM2.5 =

-79.1119 * red +

37.0308 * orange +

17.8698

La precisión de predicción del modelo se evalúa como:

r = 0.042
RMSE = 8.8081

El modelo muestra que el parámetro con mayor peso son los valores de la columna naranja. El coeficiente de correlación de este modelo es bastante bajo ($r \approx 0.042$).

4.1.2.2. Tráfico y meteorología día completo

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

PM2.5 =

16.5014 * orange +

1.4731 * temp +

2.4092 * pressure +

-1.8964 * rain +

0.2019 * RH +

-1773.3138

La precisión de predicción del modelo se evalúa como:

r = 0.407
RMSE = 8.015

El modelo muestra que el parámetro con mayor peso, son los valores de la columna naranja. El coeficiente de correlación de este modelo es más alto comparado con el anterior modelo (r≈0.407), se nota un crecimiento en el coeficiente de correlación gracias a los datos meteorológicos.

4.1.2.3. Tráfico, meteorología y químicos día completo

El modelo de regresión lineal obtenido después de ejecutar el algoritmo es:

PM2.5 =

9.0458 * CO +

0.1521 * RH +

0.3147 * NO2 +

0.2013 * O3 +

-0.004 * SolarRad +

0.7434 * SO2 +

0.7083 * temp +

0.9069 * WS +

-23.7983

La precisión de predicción del modelo se evalúa como:

r = 0.8024
RMSE = 5.2183

El modelo muestra que el parámetro con mayor peso son los valores de la columna CO. El coeficiente de correlación de este modelo es más alto que los anteriores modelos (r≈0.8024), se nota un crecimiento en el coeficiente de correlación gracias a los datos meteorológicos y químicos.

Tabla 2

Coeficientes de correlación de los modelos obtenidos aplicando el algoritmo de Regresión Lineal.

	r	RMSE	Parámetro con mayor peso
Tráfico día completo	0.042	8.8081	orange = 37.0308
Tráfico + datos meteorológicos día completo	0.407	8.015	orange = 16.5014
Tráfico + datos meteorológicos + químicos día completo	0.8024	5.2183	CO = 9.0458

Conclusión

Se observa que se realizaron tres tipos de análisis esto se debe a que al trabajar con imágenes de áreas circulares se obtuvo valores bastantes bajos como el coeficiente de correlación de 0.042, por lo que se decidió ya no trabajar con las imágenes mencionadas. En la tabla 2 donde se presentan los coeficientes de

correlación de los modelos obtenidos aplicando el algoritmo de Regresión Lineal se puede ver que el r más alto obtenido fue de 0.8024 trabajando con los datos del tráfico + datos meteorológicos + químicos en todo el día. Al crear un modelo solo con datos del tráfico se obtuvo un r demasiado bajo lo cual indica que el tráfico por sí solo no es un factor significativo en la predicción de la calidad del aire y al crear el modelo basado en el tráfico + datos meteorológicos + químicos se puede indicar que los datos meteorológicos y químicos son factores que mejoran los resultados en la predicción de la calidad del aire.

4.1.3. Tercer método: Información de imágenes de tráfico de diferentes áreas rectangulares + datos meteorológicos aplicando el algoritmo Regresión Lineal y clasificadores en Weka

Tabla 3

Coeficientes de correlación de los modelos obtenidos aplicando el algoritmo de Regresión Lineal.

	Área (km2)									
	2.2		5.0		8.7		10.9		14.0	
	r	RMSE	r	RMSE	R	RMSE	r	RMSE	r	RMSE
Tráfico día completo	0.28	8.6	0.29	8.58	0.38	8.98	0.31	8.55	0.3	8.5309
Tráfico + datos meteorológicos día completo	0.49	7.8	0.51	7.69	0.51	7.72	0.52	7.66	0.52	7.6245
Tráfico en la mañana	0.38	11.14	0.37	11.1831	0.4	11.043	0.42	10.9171	0.43	10.8608
Tráfico + datos meteorológicos en la mañana	0.62	9.41	0.62	9.44	0.63	9.37	0.64	9.32	0.65	9.1698

El coeficiente de correlación (r) más alto obtenido fue de 0.65 con los datos del tráfico + datos meteorológicos en horas de la mañana en un área de 14.0 km^2. El menor coeficiente de correlación obtenido fue de 0.28 solamente con los datos del tráfico en día completo. Se puede observar en la Tabla 3, que se obtienen mejores resultados en horas de la mañana y en áreas más grandes.

Tabla 4

Coeficientes de correlación de los modelos obtenidos en los tres métodos, aplicando el algoritmo de Regresión Lineal.

	Primer método	Segundo método	Tercer método				
			Área (km2)				
			2.2	5.0	8.7	10.9	14.0
	r	r	r	r	r	r	r
Tráfico día completo	0.2625	0.042	0.28	0.29	0.38	0.31	0.3
Tráfico + datos meteorológicos día completo	0.4084	0.407	0.49	0.51	0.51	0.52	0.52
Tráfico en la mañana	0.2899	x	0.38	0.37	0.4	0.42	0.43
Tráfico + datos meteorológicos en la mañana	0.4571	x	0.62	0.62	0.63	0.64	0.65

En la Tabla 4 se observan los coeficientes de correlación de los modelos obtenidos en los tres métodos, aplicando el algoritmo de Regresión Lineal. El coeficiente de correlación (r) más alto obtenido fue de 0.65 con los datos del tráfico + datos meteorológicos en horas de la mañana en un área de 14.0 km^2. El menor coeficiente de correlación obtenido fue de 0.2625 solamente con los datos del tráfico en día completo. Se puede observar en la Tabla 4, que se obtienen mejores resultados en el tercer método donde se trabajó con diferentes áreas.

4.2 Evaluación del rendimiento de las técnicas de minería de datos utilizados

4.2.1. Evaluación de la Regresión

Lo siguiente es en base a los resultados obtenidos en el tercer método, se trabaja solo con los resultados mencionados debido a que se obtuvieron mejores valores así que el primer y segundo método quedaron descartados.

Se genera una matriz de confusión la cual se observa en la Figura 31 , con los datos del tráfico, PM2.5 real, PM2.5 predictivo, PM2.5 real binario y PM2.5 predictivo binario en día completo en un área de 2.2 km^2. Las columnas PM2.5 real binario y PM2.5 predictivo binario son valores binarios; para obtener dichos valores, se obtuvo la mediana de PM2.5 real y así según una condición es 1 si es mayor y 0 si es menor a la mediana obtenida. Con los valores de PM2.5 real binario y PM2.5 predictivo binario se obtiene el número de valores de cada cuadrante de la matriz de confusión, VN (VERDADEROS NEGATIVOS), FN (FALSOS NEGATIVOS), FP (FALSOS POSITIVOS) y VP (VERDADEROS POSITIVOS). Finalmente se calcula el porcentaje (%) de desempeño, que es igual a 59.64.

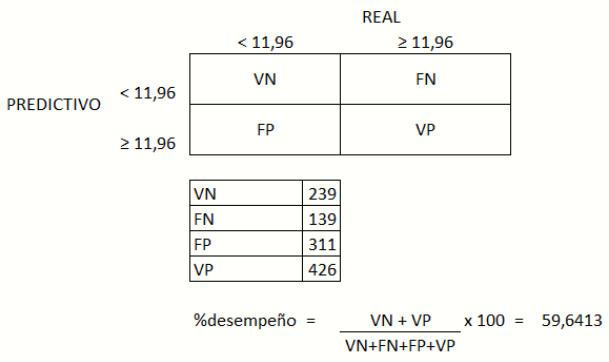

Figura 13. Matriz de confusión del tráfico en día completo en un área de 2.2 km^2.

Tabla 5

Coeficientes de correlación obtenidos en los modelos aplicando el algoritmo de Regresión Lineal.

	Área (km2)				
	2.2	5.0	8.7	10.9	14.0
	% desempeño	% desempeño	% desempeño	% desempeño	% desempeño
Tráfico día completo	59.6413	59.93	59.84	59.42	59.42
Tráfico + datos meteorológicos día completo	70.43	70.05	69.77	70.36	71.29
Tráfico en la mañana	61.78	61.78	61.78	64.00	62.83
Tráfico + datos meteorológicos en la mañana	75.80	75.34	74.43	74.43	75.91

El mejor porcentaje de desempeño fue de 75.91%, se obtuvo en el modelo basado en tráfico + datos meteorológicos en horas de la mañana en un área de 14.0 km^2 cómo se observa en la Tabla 5, se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de concentración. El menor porcentaje de desempeño fue de 59.42% solamente con los datos del tráfico en día completo.

4.2.2. Evaluación de la Clasificación

Kappa mide el grado de concordancia de las evaluaciones nominales u ordinales realizadas por múltiples evaluadores cuando se evalúan las mismas muestras.

Los valores de kappa varían de –1 a +1. Mientras más alto sea el valor de kappa, más fuerte será la concordancia. Cuando:

- Kappa = 1, existe concordancia perfecta.
- Kappa = 0, la concordancia es la misma que se esperaría en virtud de las probabilidades.
- Kappa < 0, la concordancia es más débil que lo esperado en virtud de las probabilidades; esto casi nunca sucede.

Cuando sus clasificaciones sean nominales (verdadero/falso, bueno/malo, crujiente/crocante/blando), utilice los estadísticos kappa. (Minitab, 2017)

4.2.2.1. Resultados de los modelos obtenidos aplicando KNN en diferentes áreas

Tabla 6

Coeficientes kappa obtenidos en los modelos generados en Weka aplicando KNN.

	Área (km2)									
	2.2		5.0		8.7		10.9		14.0	
	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE
Tráfico día completo	0.0672	0.6822	0.0504	0.6883	0.0543	0.687	0.0658	0.6828	0.0739	0.6798
Tráfico + datos meteorológicos día completo	0.3471	0.5708	0.3283	0.5789	0.3609	0.5647	0.3652	0.5628	0.3443	0.5721
Tráfico en la mañana	0.1011	0.6667	0.1011	0.6667	-0.0216	0.7114	-0.0661	0.7267	0.1534	0.6472
Tráfico + datos meteorológicos en la mañana	0.3769	0.5551	0.3822	0.5526	0.3771	0.5551	0.3768	0.5551	0.3125	0.5834

El mejor coeficiente de kappa obtenido fue de 0.3822 en el modelo basado en tráfico + datos meteorológicos en horas de la mañana en un área de 5.0 km^2, el peor coeficiente de kappa fue de -0.0661 cómo se observa en la Tabla 6, se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de

4.2.2.2. Resultados de los modelos obtenidos aplicando Regresión Logística

Tabla 7

Coeficientes kappa obtenidos en los modelos generados en Weka aplicando Regresión Logística.

	Área (km2)									
	2.2		5.0		8.7		10.9		14.0	
	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE
Tráfico día completo	0.2083	0.4868	0.2088	0.4864	0.2123	0.4861	0.2165	0.4858	0.2377	0.4862
Tráfico + datos meteorológicos día completo	0.4866	0.4309	0.4527	0.4281	0.4564	0.428	0.4694	0.4288	0.4573	0.4301
Tráfico en la mañana	0.3054	0.4809	0.3054	0.4809	0.2874	0.4776	0.2964	0.4777	0.2855	0.4803
Tráfico + datos meteorológicos en la mañana	0.564	0.408	0.5322	0.4107	0.5552	0.4067	0.5197	0.409	0.5413	0.4069

El mejor coeficiente de kappa obtenido fue de 0.552 en el modelo basado en tráfico en horas de la mañana en un área de 8.7 km^2, el peor coeficiente de kappa fue de 0.2083 cómo se observa en la Tabla 7, se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de concentración.

4.2.2.3. Resultados de los modelos obtenidos aplicando Redes Neuronales

Tabla 8

Coeficientes kappa obtenidos en los modelos aplicando Redes Neuronales.

	Área (km2)									
	2.2		5.0		8.7		10.9		14.0	
	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE
Tráfico día completo	0.1905	0.4917	0.2124	0.4899	0.2234	0.4887	0.2033	0.4927	0.1905	0.4886
Tráfico + datos meteorológicos día completo	0.386	0.4326	0.4924	0.4316	0.5032	0.433	0.4905	0.4383	0.4699	0.4389
Tráfico en la mañana	0.2341	0.4835	0.2341	0.4835	0.2433	0.4801	0.2257	0.4901	0.243	0.4859
Tráfico + datos meteorológicos en la mañana	0.4574	0.4649	0.3559	0.5302	0.5149	0.4954	0.4216	0.4836	0.3999	0.4985

El mejor coeficiente de kappa obtenido fue de 0.5149 en el modelo basado en tráfico en horas de la mañana en un área de 8.7 km^2, el peor coeficiente de kappa fue de 0.1905 cómo se observa en la Tabla8 , se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de concentración.

4.2.2.4. Resultados de los modelos obtenidos aplicando SVM

Tabla 9

Coeficientes kappa obtenidos en los modelos aplicando SVM.

	Área (km2)									
	2.2		5.0		8.7		10.9		14.0	
	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE	CK	RMSE
Tráfico día completo	0.2103	0.6289	0.2176	0.6259	0.2138	0.6250	0.2218	0.6241	0.2314	0.6202
Tráfico + datos meteorológicos día completo	0.4665	0.5161	0.4813	0.5089	0.4869	0.5063	0.4782	0.5105	0.4842	0.5076
Tráfico en la mañana	0.2525	0.611	0.2525	0.611	0.2608	0.6074	0.2696	0.6037	0.188	0.6366
Tráfico + datos meteorológicos en la mañana	0.5277	0.4853	0.5214	0.4877	0.5366	0.4807	0.5365	0.4807	0.5586	0.4693

El mejor coeficiente de kappa obtenido fue de 0.5586 en el modelo basado en tráfico en horas de la mañana en un área de 14.0 km^2, el peor coeficiente de kappa fue de 0.188 cómo se observa en la Tabla 9 , se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de concentración.

Tabla 10

Resumen de los coeficientes kappa obtenidos en los modelos aplicando SVM, Regresión Logística, KNN y Redes Neuronales.

	Área (km2)																			
	2.2				5.0				8.7				10.9				14.0			
	CK				CK				CK				CK				CK			
	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN
Tráfico día completo	0.21	0.21	0.07	0.19	0.22	0.21	0.05	0.21	0.21	0.21	0.05	0.22	0.22	0.22	0.07	0.20	0.23	0.24	0.07	0.19
Tráfico + datos meteorológicos día completo	0.47	0.49	0.35	0.39	0.48	0.45	0.33	0.49	0.49	0.46	0.36	0.50	0.48	0.47	0.37	0.49	0.48	0.46	0.34	0.47
Tráfico en la mañana	0.25	0.31	0.10	0.23	0.25	0.31	0.10	0.23	0.26	0.29	-0.02	0.24	0.27	0.29	-0.07	0.22	0.18	0.29	0.15	0.24
Tráfico + datos meteorológicos en la mañana	0.53	0.56	0.38	0.46	0.52	0.53	0.38	0.36	0.54	0.56	0.38	0.51	0.54	0.52	0.38	0.42	0.56	0.54	0.31	0.40

El mejor coeficiente de kappa obtenido entre todos los modelos fue de 0.56 en el modelo basado en tráfico + datos meteorológicos en horas de la mañana en un área de 8.7 km^2, el peor coeficiente de kappa fue de 0.07 cómo se observa en la Tabla 10, se debe a que entre las horas entre 6am a 10am existe más tráfico y los datos meteorológicos presentan mayores niveles de concentración.

5. Capítulo V: Análisis de los resultados

5.1. Análisis infográfico e interpretativo de los principales resultados de clasificación

El archivo cargado contiene los datos sobre el tráfico del día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa que en la Figura 14 las curvas ROC de los clasificadores SVM yregresión logística son un test excelente con un área bajo la curva (AUC) de 1. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.74 menor al área bajo la curva de los clasificadores SVM y regresión logística pero mayor al clasificador KNN que es de 0.56. La curva ROC del clasificador KNN es un test malo. En el Anexo 4 se presenta el código completo de la creación de curvas ROC.

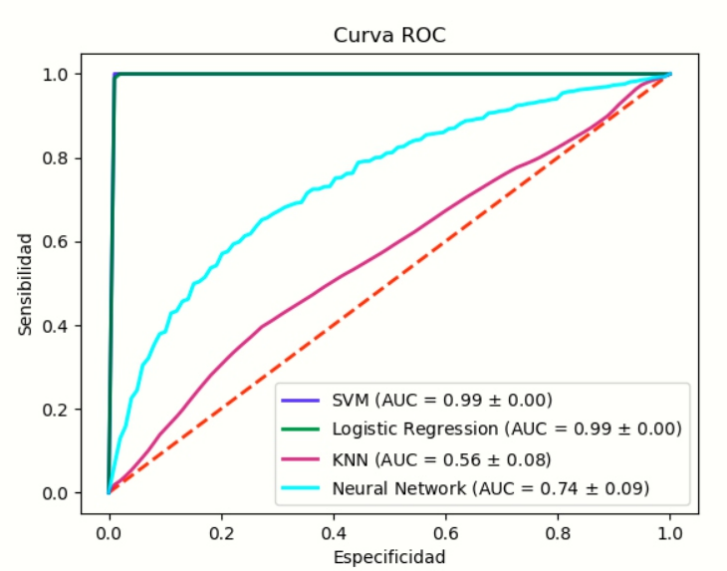

Figura 14. Curva ROC tráfico día completo en un área de 2.2 km^2.

El archivo cargado contiene los datos sobre el tráfico del día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5real de un área de 2.2 km^2. Se observa que en la Figura

15 las curvas ROC de los clasificadores SVM y regresión logística con un test regular con un área bajo la curva (AUC) de 0.69. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.55 menor al área bajo la curva de los clasificadores SVM, regresión logística y KNN que es de 0.74. La curva ROC del clasificador KNN es un test regular.

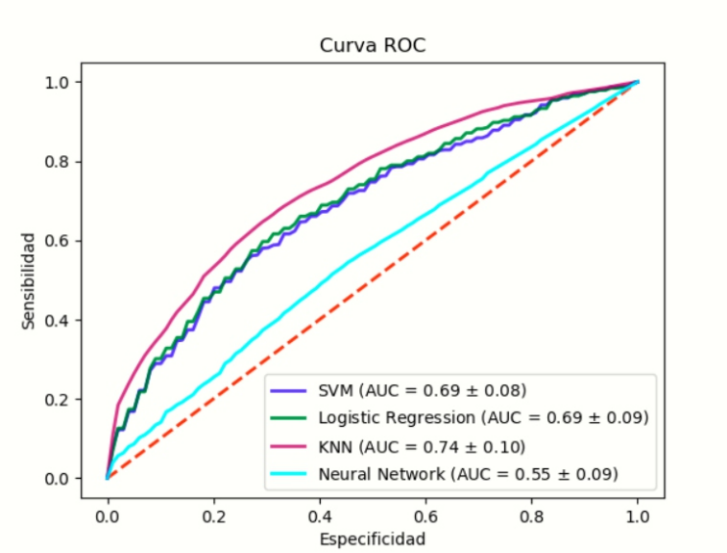

Figura 15. Curva ROC tráfico y meteorología día completo en un área de 2.2 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa que en la Figura 16 las curvas ROC de los clasificadores SVM con AUC = 0.68 y regresión logística con AUC = 0.67 son test regulares. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.64 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.61. La curva ROC del clasificador KNN es un test regular.

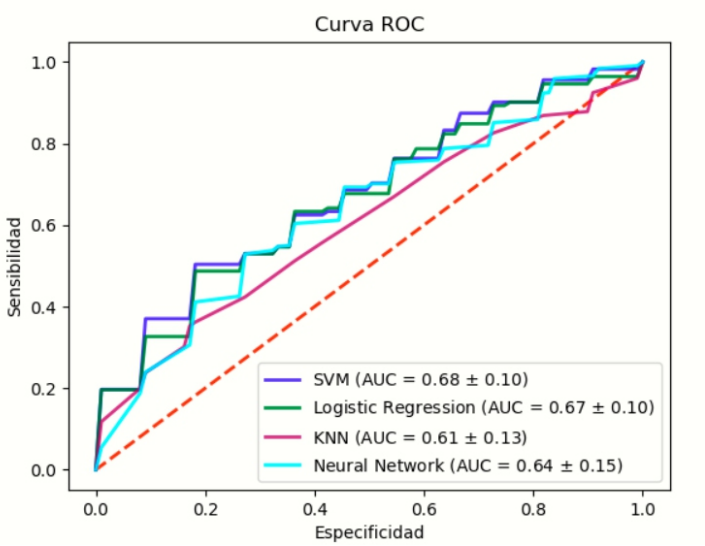

Figura 16. Curva ROC tráfico en la mañana en un área de 2.2 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 2.2km^2. Se observa en la Figura 17 las curvas ROC de los clasificadores SVM con AUC = 0.71 y regresión logística con AUC = 0.72 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.51 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.71. La curva ROC del clasificador KNN es un test regular.

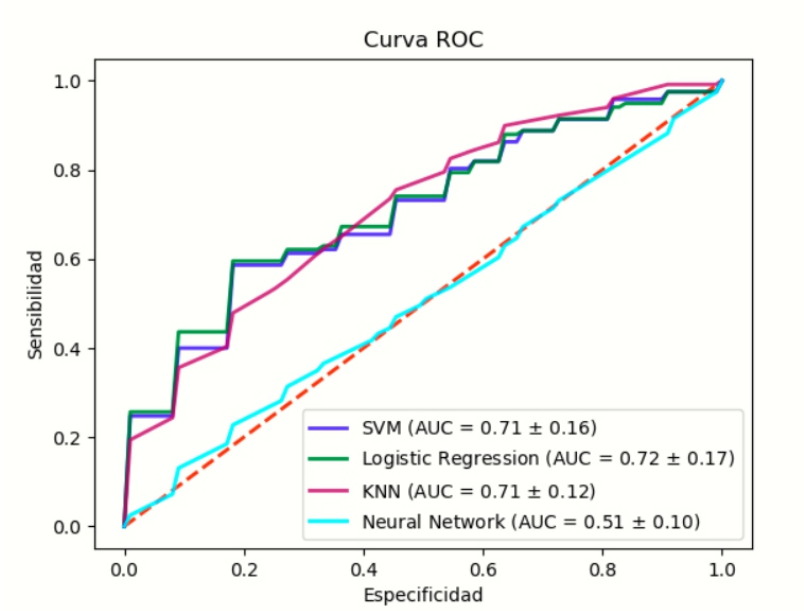

Figura 17. Curva ROC tráfico y meteorología en la mañana en un área de 2.2 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 5.0km^2. Se observa en la Figura1 8 las curvas ROC de los clasificadores SVM con AUC = 1 y regresión logística con AUC = 1 son test excelentes. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.73 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.55. La curva ROC del clasificador KNN es un test malo.

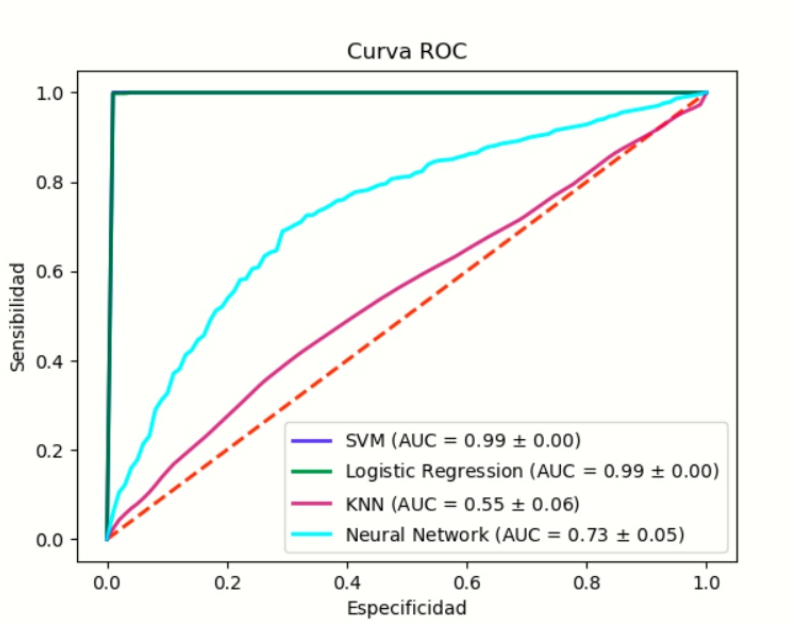

Figura 18. Curva ROC tráfico día completo en un área de 5.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 19 las curvas ROC de los clasificadores SVM con AUC = 0.74 y regresión logística con AUC = 0.74 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.52 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.74. La curva ROC del clasificador KNN es un test regular.

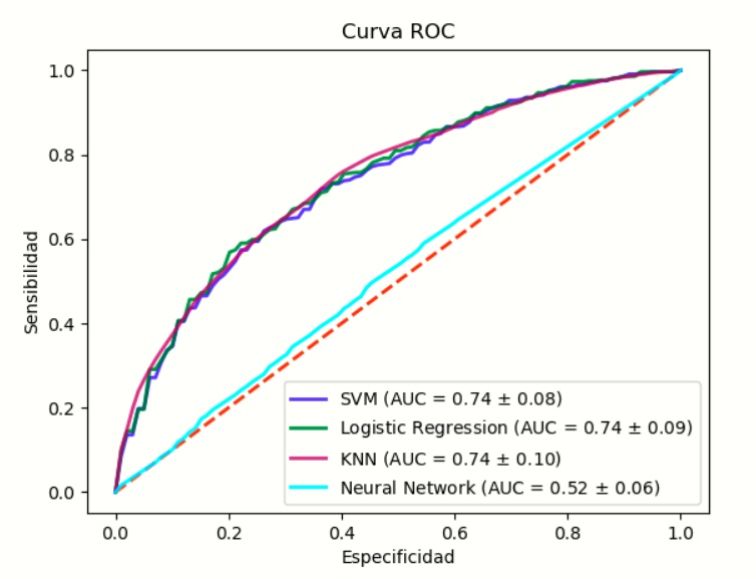

Figura 19. Curva ROC tráfico y meteorología día completo en un área de 5.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 20 las curvas ROC de los clasificadores SVM con AUC =0.68 y regresión logística con AUC = 0.67 son test regulares. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.64 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.61. La curva ROC del clasificador KNN es un test regular.

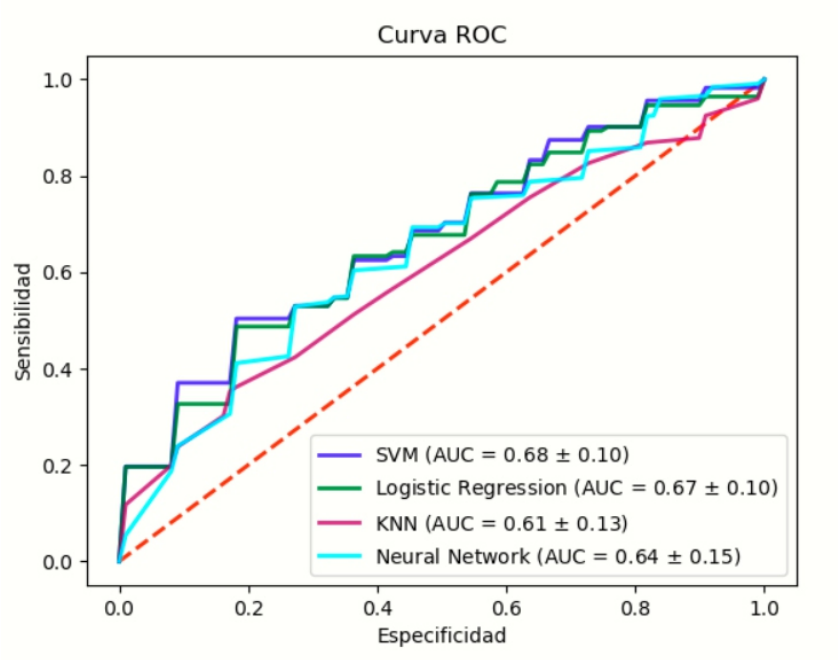

Figura 20. Curva ROC tráfico en la mañana en un área de 5.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 21 las curvas ROC de los clasificadores SVM con AUC = 0.71 y regresión logística con AUC = 0.72 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.53 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.71. La curva ROC del clasificador KNN es un test regular.

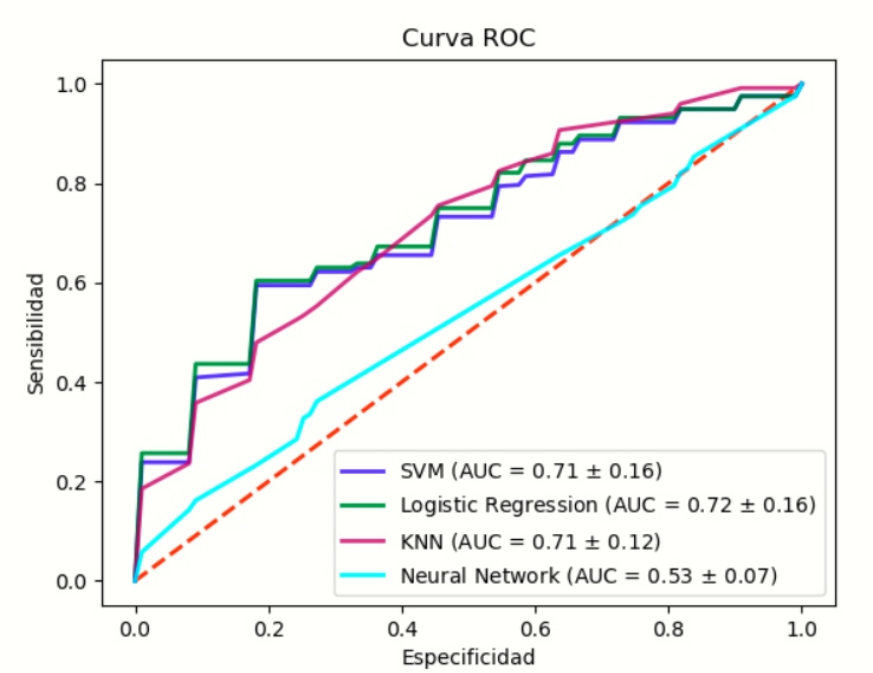

Figura 21. Curva ROC tráfico y meteorología en la mañana en un área de 5.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 8.7km^2. Se observa en la Figura2 2 las curvas ROC de los clasificadores SVM con AUC = 1 y regresión logística con AUC = 1 son test excelentes. La curva ROC del clasificador red neuronal es un test bueno, su área bajo la curva es de 0.75 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.55. La curva ROC del clasificador KNN es un test malo.

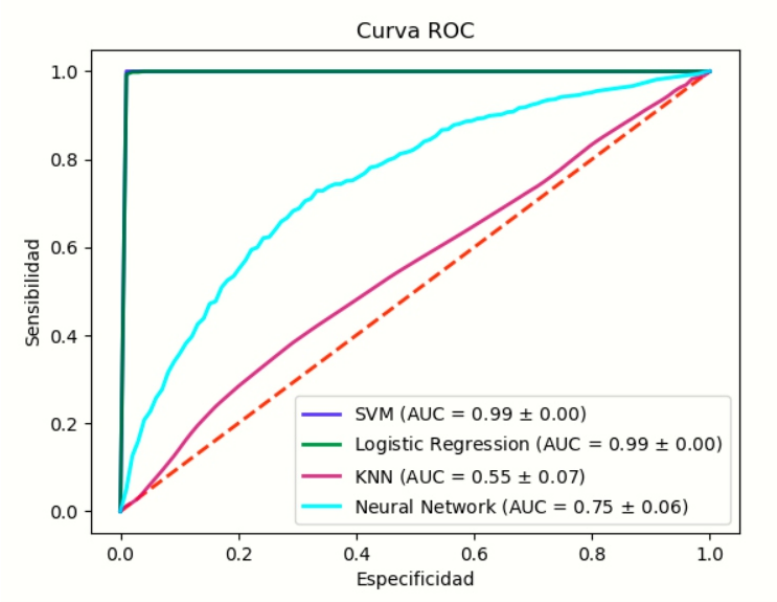

Figura 22. Curva ROC tráfico día completo en un área de 8.7 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 8.7 km^2. Se observa en la Figura 23 las curvas ROC de los clasificadores SVM con AUC = 0.75 y regresión logística con AUC = 0.75 son test buenos. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.53 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.74. La curva ROC del clasificador KNN es un test regular.

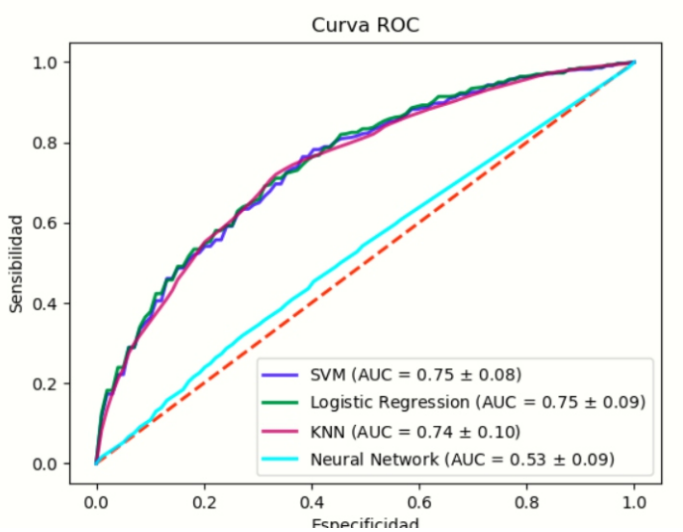

Figura 23. Curva ROC tráfico y meteorología día completo en un área de 8.7 km^2.

El archivo cargado contiene los datos sobre el tráfico en la mañana en los colores rojo y naranja, adicional el valor binario de PM2.5real de un área de 8.7 km^2. Se observa en la Figura 24 las curvas ROC de los clasificadores SVM con AUC = 0.68 y regresión logística con AUC = 0.67 son test regulares. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.63 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.61. La curva ROC del clasificador KNN es un test regular.

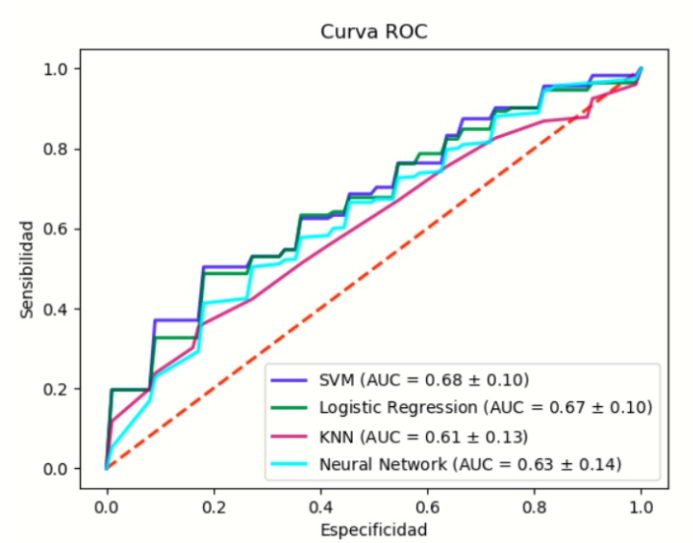

Figura 24. Curva ROC en la mañana en un área de 8.7 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 8.7km^2. Se observa en la Figura 25 las curvas ROC de los clasificadores SVM con AUC = 0.71 y regresión logística con AUC = 0.72 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.54 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.71. La curva ROC del clasificador KNN es un test regular.

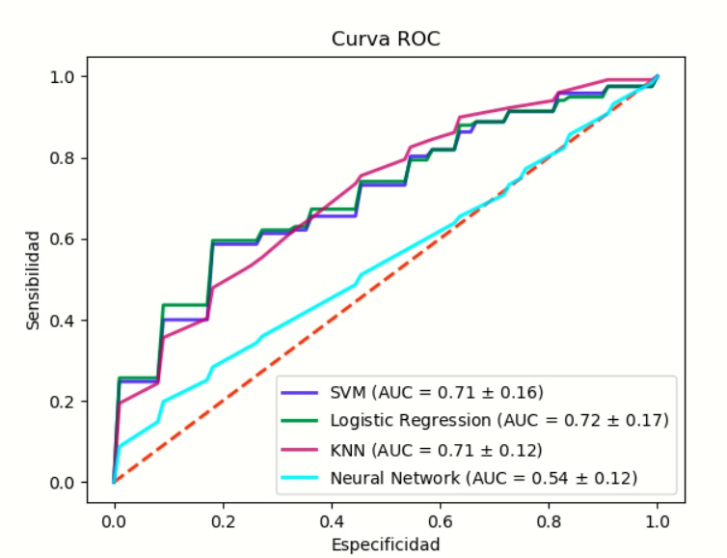

Figura 25. Curva ROC tráfico y meteorología en la mañana en un área de 8.7 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 26 las curvas ROC de los clasificadores SVM con AUC = 1 y regresión logística con AUC = 1 son test excelentes. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.72 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.55. La curva ROC del clasificador KNN es un test malo.

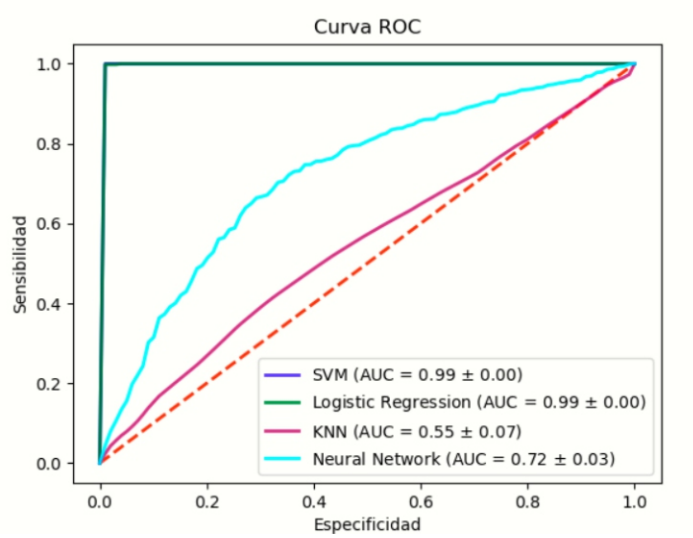

Figura 26. Curva ROC tráfico día completo en un área de 10.9 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 27 las curvas ROC de los clasificadores SVM con AUC = 0.74 y regresión logística con AUC = 0.74 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.50 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.75. La curva ROC del clasificador KNN es un test bueno.

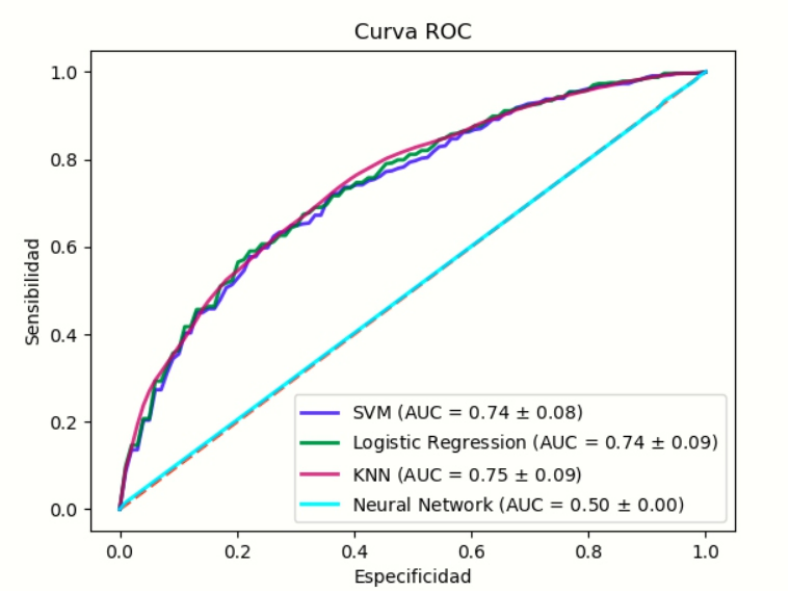

Figura 27. Curva ROC tráfico y meteorología día completo en un área de 10.9 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 28 las curvas ROC de los clasificadores SVM con AUC = 0.68 y regresión logística con AUC = 0.67 son test regulares. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.64 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.61. La curva ROC del clasificador KNN es un test regular.

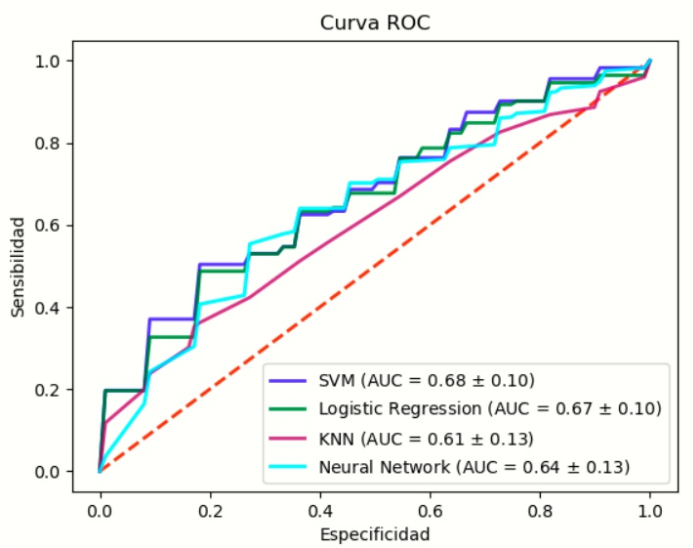

Figura 28. Curva ROC tráfico en la mañana en un área de 10.9 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 29 las curvas ROC de los clasificadores SVM con AUC = 0.71 y regresión logística con AUC = 0.72 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.51 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.71. La curva ROC del clasificador KNN es un test regular.

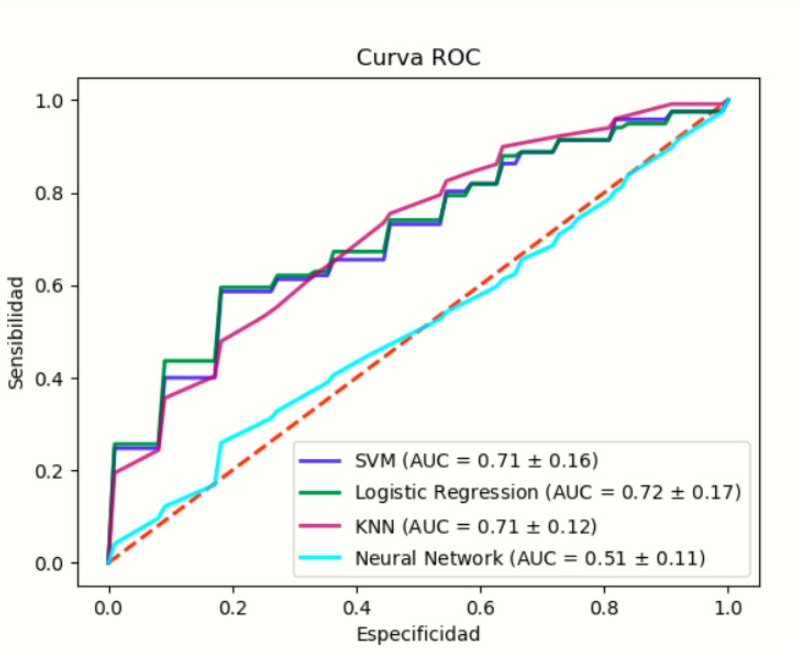

Figura 28. Curva ROC tráfico y meteorología en la mañana en un área de 10.9 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 30 las curvas ROC de los clasificadores SVM con AUC = 1 y regresión logística con AUC = 1 son test excelentes. La curva ROC del clasificador red neuronal es un test regular, su área bajo la curva es de 0.74 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.55. La curva ROC del clasificador KNN es un test malo.

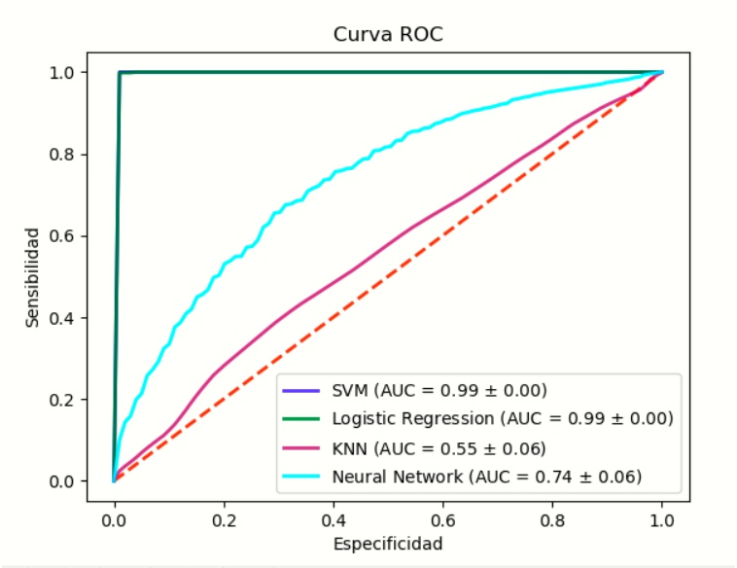

Figura 30. Curva ROC tráfico día completo en un área de 14.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 31 las curvas ROC de los clasificadores SVM con AUC = 0.72 y regresión logística con AUC = 0.73 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.58 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.75. La curva ROC del clasificador KNN es un test bueno.

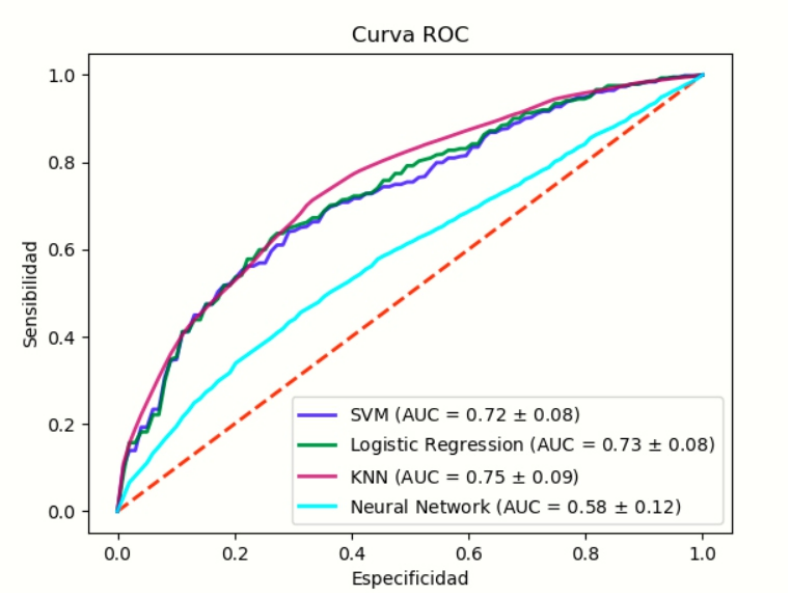

Figura 31. Curva ROC tráfico y meteorología día completo en un área de 14.0 km^2.

El archivo cargado contiene los datos sobre el tráfico en el mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 32 las curvas ROC de los clasificadores SVM con AUC = 0.57 y regresión logística con AUC = 0.58 son test malos. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.52 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.56. La curva ROC del clasificador KNN es un test malo.

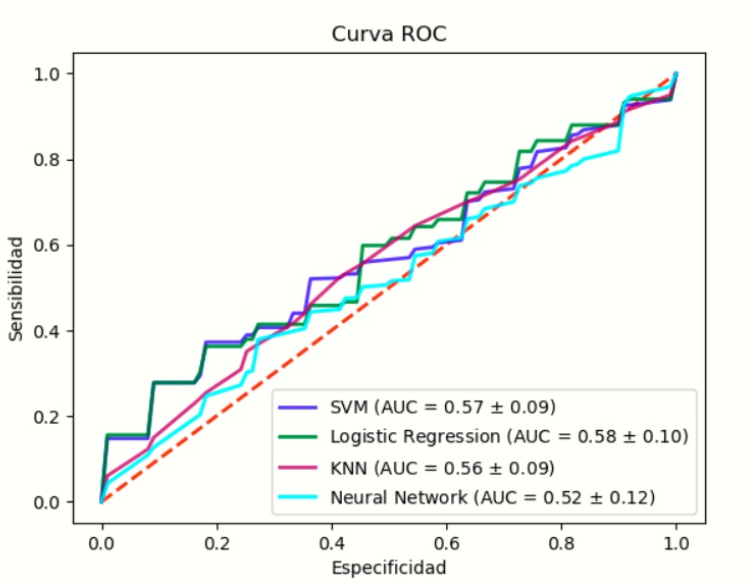

Figura 32. Curva ROC tráfico en la mañana en un área de 14.0 km^2.

El archivo cargado contiene los datos sobre el tráfico y meteorología en el mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 33 las curvas ROC de los clasificadores SVM con AUC = 0.69 y regresión logística con AUC = 0.70 son test regulares. La curva ROC del clasificador red neuronal es un test malo, su área bajo la curva es de 0.55 menor al área bajo la curva de los clasificadores SVM y regresión logística. La curva ROC del clasificador KNN es de 0.72. La curva ROC del clasificador KNN es un test regular.

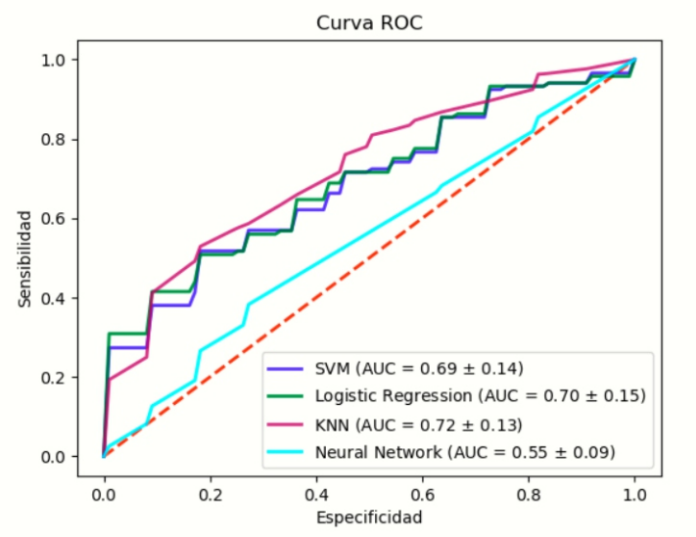

Figura 33. Curva ROC tráfico y meteorología en la mañana en un área de 14 km^2.

Tabla 11

Valores del área bajo la curva de todos los modelos aplicando SVM, Regresión Logística, KNN y Redes Neuronales.

	Área (km2)																			
	2.2				5.0				8.7				10.9				14.0			
	AUC				AUC				AUC				AUC				AUC			
	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN	SVM	RL	KNN	RN
Tráfico día completo	0.99	0.99	0.56	0.74	0.99	0.99	0.55	0.73	0.99	0.99	0.55	0.75	0.99	0.99	0.55	0.72	0.99	0.99	0.55	0.74
Tráfico + datos meteorológicos día completo	0.69	0.69	0.74	0.55	0.74	0.74	0.74	0.52	0.75	0.75	0.74	0.53	0.74	0.74	0.75	0.50	0.72	0.73	0.75	0.58
Tráfico en la mañana	0.68	0.67	0.61	0.64	0.68	0.67	0.61	0.64	0.68	0.67	0.61	0.63	0.68	0.67	0.61	0.64	0.57	0.58	0.56	0.52
Tráfico + datos meteorológicos en la mañana	0.71	0.72	0.71	0.51	0.71	0.72	0.71	0.53	0.71	0.72	0.71	0.54	0.71	0.72	0.71	0.51	0.69	0.70	0.72	0.55

El mejor valor del área bajo la curva fue de 0.99 se obtuvo en SVM, con los datos del tráfico en día completo en las diferentes áreas. El peor valor del área bajo la curva fue de 0.50 como se observa en la Tabla 11. El valor más bajo del área bajo la curva se obtuvo en Redes Neuronales con los datos del tráfico + datos meteorológicos en un área de 10.9 km^2.

5.2. Análisis infográfico e interpretativo de los principales resultados de regresión

Coeficientes de correlación del tráfico en la mañana en distintas áreas como se observa en la Tabla 12, los resultados fueron obtenidos al ejecutar el algoritmo de regresión lineal en Weka.

Tabla 12

Coeficientes de correlación en cada área

Área (km^2)	r
2.2	0.38
5.0	0.3746
8.7	0.3995
10.9	0.4226
14.0	0.4262

Se observa en la Figura 34 que el coeficiente de correlación aumenta conforme aumenta el área que se evalúa.

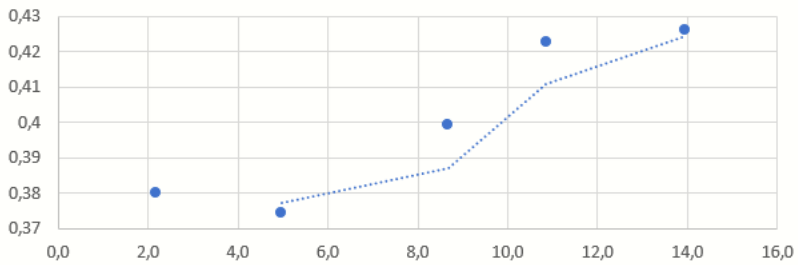

Figura 34. Gráfica con los coeficientes de correlación en cada área.

Coeficientes de correlación del tráfico y meteorología en la mañana en distintas áreas como se observa en la Tabla 13.

Tabla 13

Coeficientes de correlación en cada área

Área (km^2)	r
2.2	0.6262
5.0	0.6235
8.7	0.6302
10.9	0.6353
14.0	0.6467

Se observa en la Figura 35 que el coeficiente de correlación aumenta conforme aumenta el área que se evalúa, se obtienen coeficientes de correlación mayores gracias a los datos meteorológicos.

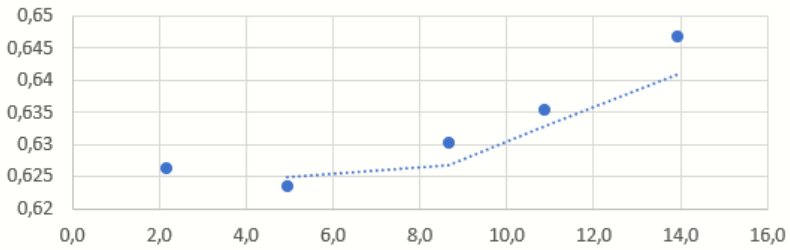

Figura 35. Gráfica con los coeficientes de correlación en cada área.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa en la Figura 36 la curva ROC de la regresión lineal con AUC = 0.55 es un test malo.

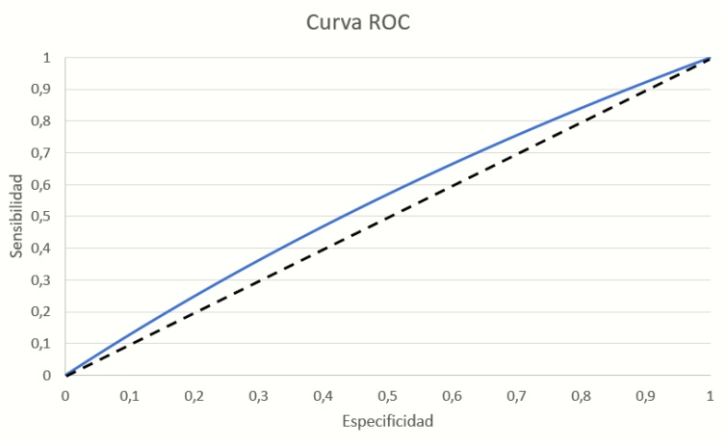

Figura 36. Curva ROC tráfico día completo en un área de 2.2 km^2 con AUC = 0.55.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa en la Figura 37 la curva ROC de la regresión lineal con AUC = 0.64 es un test regular.

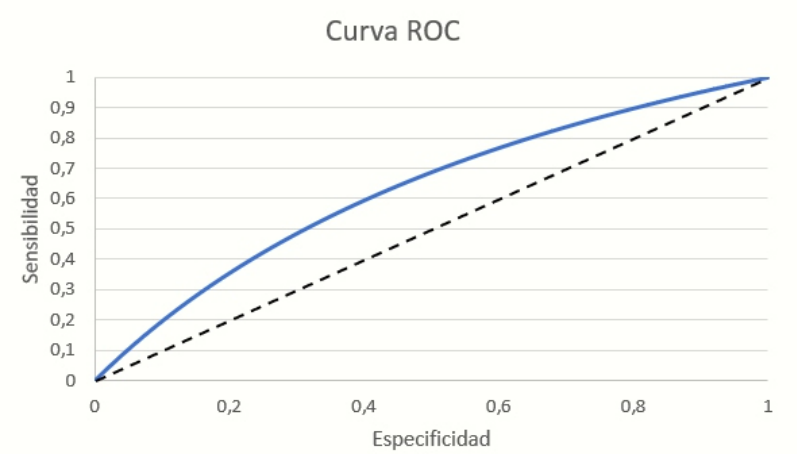

Figura 37. Curva ROC tráfico y meteorología día completo en un área de 2.2 km^2 con AUC = 0.64.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa en la Figura 38 la curva ROC de la regresión lineal con AUC = 0.58 es un test malo.

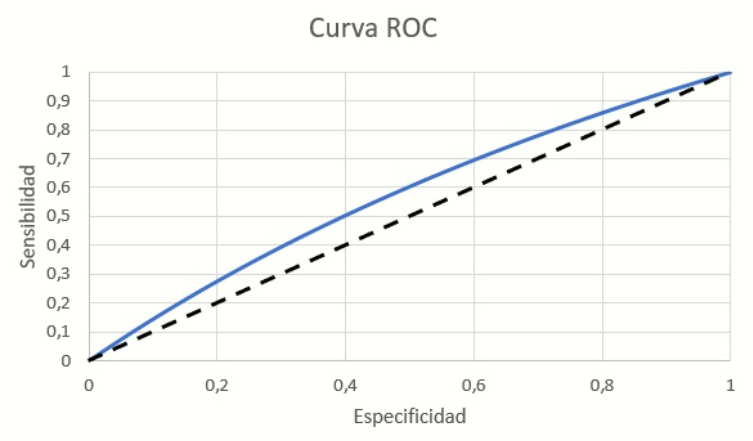

Figura 38. Curva ROC tráfico en la mañana en un área de 2.2 km^2 con AUC = 0.58.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 2.2 km^2. Se observa en la Figura 39 la curva ROC de la regresión lineal con AUC = 0.68 es un test regular.

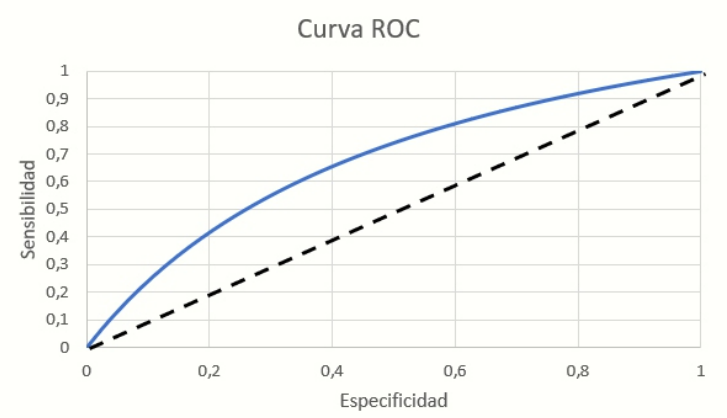

Figura 39. Curva ROC tráfico y meteorología en la mañana en un área de 2.2 km^2 con AUC = 0.68.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 40 la curva ROC de la regresión lineal con AUC = 0.56 es un test malo.

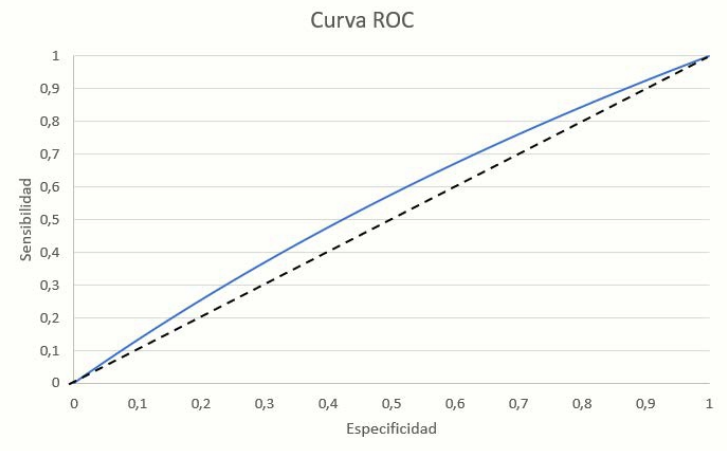

Figura 40. Curva ROC tráfico día completo en un área de 5.0 km^2 con AUC = 0.56.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 41 la curva ROC de la regresión lineal con AUC = 0.63 es un test regular.

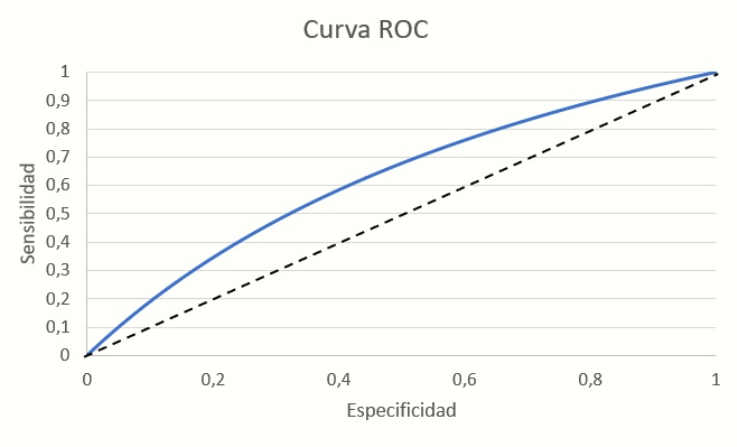

Figura 41. Curva ROC tráfico y meteorología día completo en un área de 5.0 km^2 con AUC = 0.63.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 42 la curva ROC de la regresión lineal con AUC = 0.58 es un test malo.

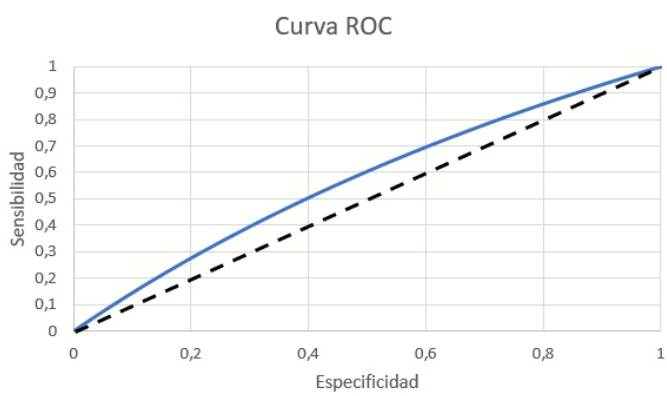

Figura 42. Curva ROC tráfico en la mañana en un área de 5.0 km^2 con AUC = 0.58.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 5.0 km^2. Se observa en la Figura 43 la curva ROC de la regresión lineal con AUC = 0.69 es un test regular.

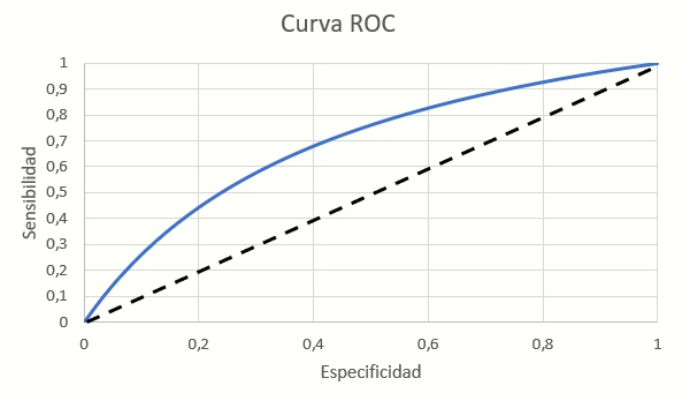

Figura 43. Curva ROC tráfico y meteorología en la mañana en un área de 5.0 km^2 con AUC = 0.69.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 8.7 km^2. Se observa en la Figura 44 la curva ROC de la regresión lineal con AUC = 0.56 es un test malo.

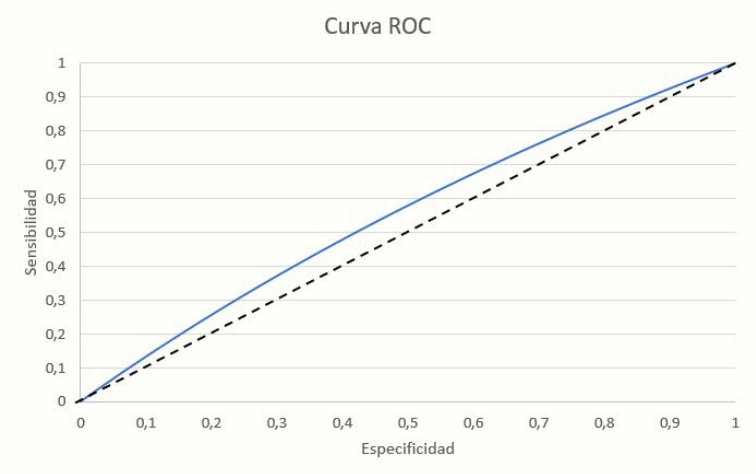

Figura 44. Curva ROC tráfico día completo en un área de 8.7 km^2 con AUC = 0.56.

El archivo cargado contiene los datos sobre el tráfico en el día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 8.7 km^2. Se observa en la Figura 45 la curva ROC de la regresión lineal con AUC = 0.63 es un test regular.

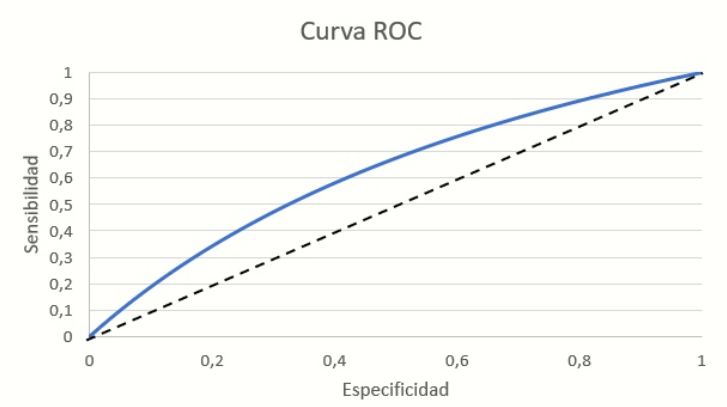

Figura 45. Curva ROC tráfico y meteorología día completo en un área de 8.7 km^2 con AUC = 0.63.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 8.7 km^2. Se observa en la Figura 46 la curva ROC de la regresión lineal con AUC = 0.58 es un test regular.

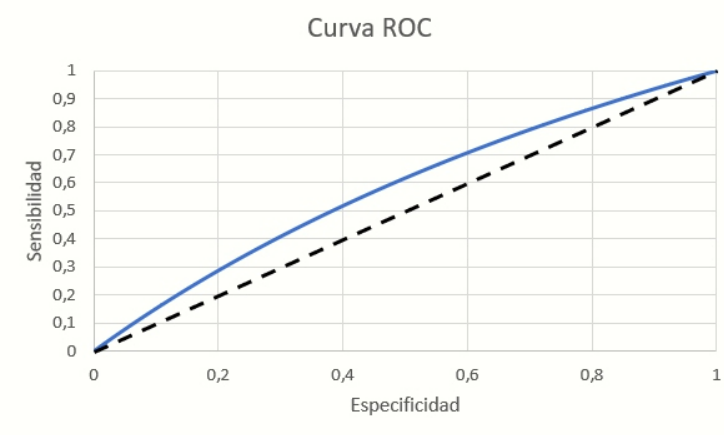

Figura 46. Curva ROC tráfico en la mañana en un área de 8.7 km^2 con AUC = 0.58.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 8.7 km^2. Se observa en la Figura 47 la curva ROC de la regresión lineal con AUC = 0.67 es un test regular.

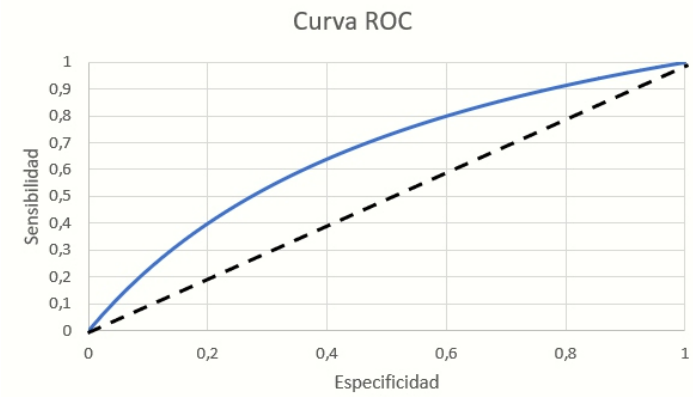

Figura 47. Curva ROC tráfico y meteorología en la mañana en un área de 8.7 km^2 con AUC = 0.67.

El archivo cargado contiene los datos sobre el tráfico en día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 48 la curva ROC de la regresión lineal con AUC = 0.56 es un test malo.

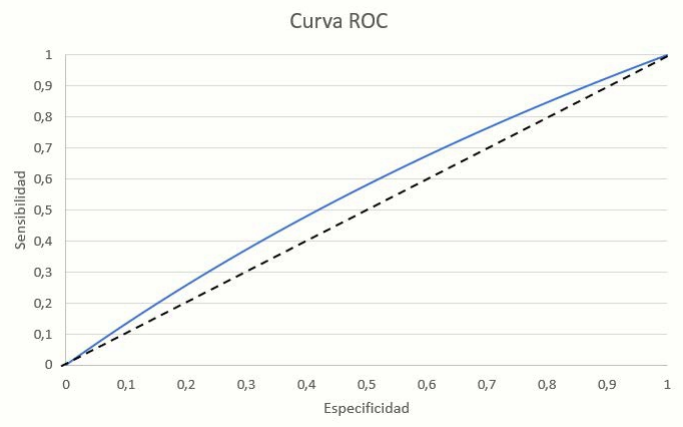

Figura 48. Curva ROC tráfico día completo en un área de 10.9 km^2 con AUC = 0.56.

El archivo cargado contiene los datos sobre el tráfico en día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 49 la curva ROC de la regresión lineal con AUC = 0.63 es un test regular.

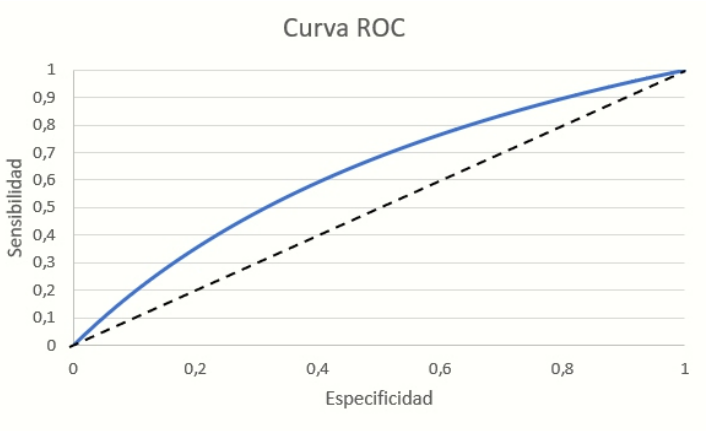

Figura 49. Curva ROC tráfico y meteorología día completo en un área de 10.9 km^2 con AUC = 0.63.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 50 la curva ROC de la regresión lineal con AUC = 0.59 es un test malo.

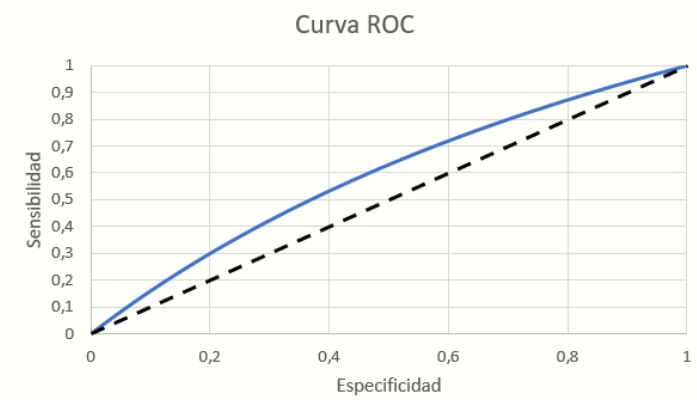

Figura 50. Curva ROC tráfico en la mañana en un área de 10.9 km^2 con AUC = 0.59.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 10.9 km^2. Se observa en la Figura 51 la curva ROC de la regresión lineal con AUC = 0.68 es un test regular.

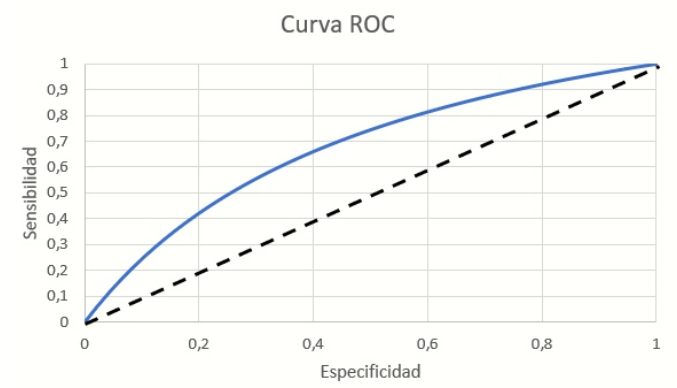

Figura 51. Curva ROC tráfico y meteorología en la mañana en un área de 10.9 km^2 con AUC = 0.68.

El archivo cargado contiene los datos sobre el tráfico en día completo representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 52 la curva ROC de la regresión lineal con AUC = 0.56 es un test malo.

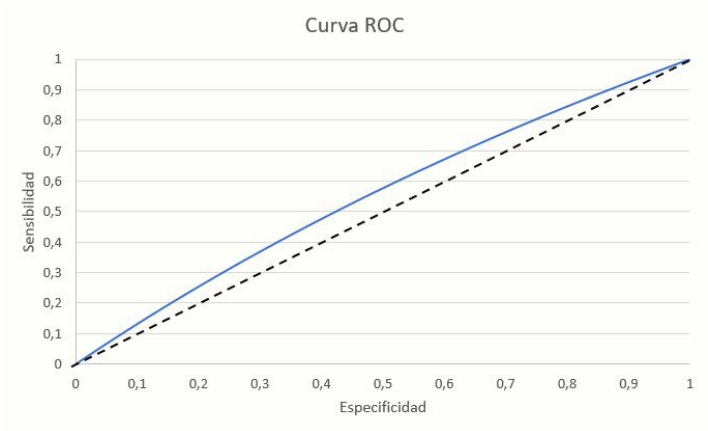

Figura 52. Curva ROC tráfico día completo en un área de 14.0 km^2 con AUC = 0.56.

El archivo cargado contiene los datos sobre el tráfico en día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 53 la curva ROC de la regresión lineal con AUC = 0.63 es un test regular.

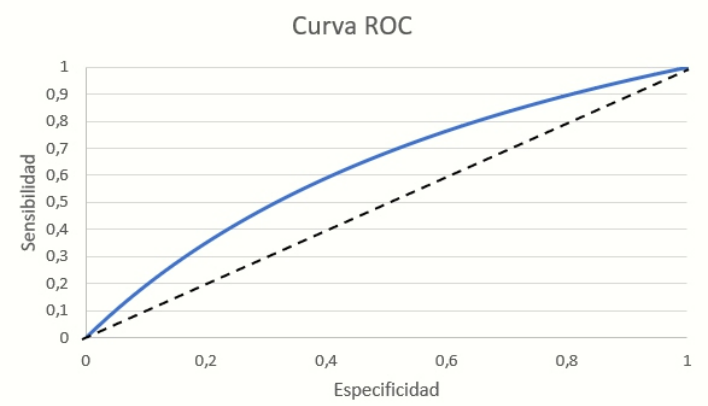

Figura 53. Curva ROC tráfico y meteorología día completo en un área de 14.0 km^2 con AUC = 0.63.

El archivo cargado contiene los datos sobre el tráfico en la mañana representado en los colores rojo y naranja, adicional el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 54 la curva ROC de la regresión lineal con AUC = 0.63 es un test regular.

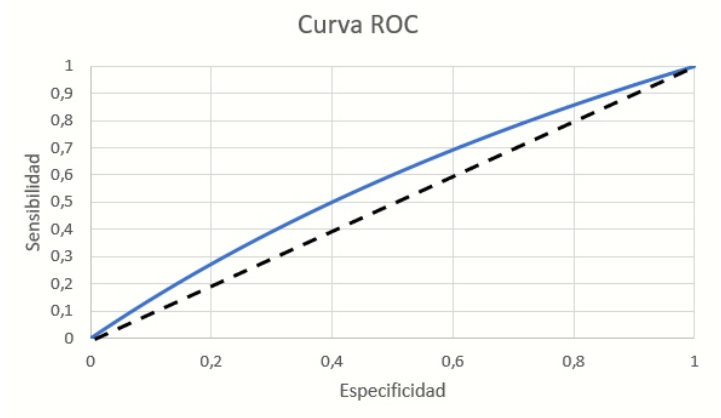

Figura 54. Curva ROC tráfico en la mañana en un área de 14.0 km^2 con AUC = 0.57.

El archivo cargado contiene los datos sobre el tráfico en día completo representado en los colores rojo y naranja, adicionales datos meteorológicos y el valor binario de PM2.5 real de un área de 14.0 km^2. Se observa en la Figura 55 la curva ROC de la regresión lineal con AUC = 0.66 es un test regular.

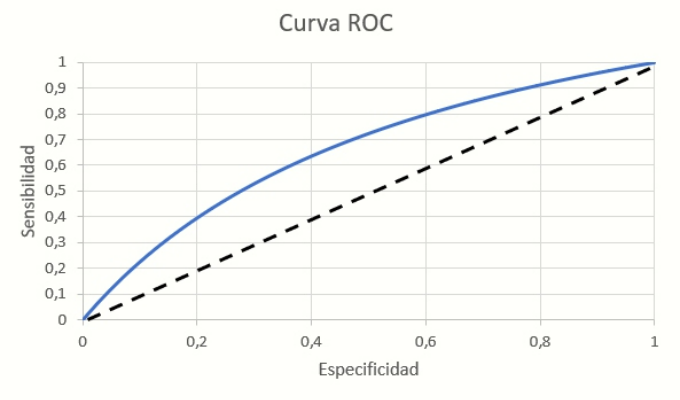

Figura 55. Curva ROC tráfico y meteorología en la mañana en un área de 14.0 km^2 con AUC = 0.66.

Tabla 14

Valores del área bajo la curva en Regresión Lineal.

	Área (km2)				
	2.2	5.0	8.7	10.9	14.0
	AUC	AUC	AUC	AUC	AUC
Tráfico día completo	0.55	0.56	0.56	0.56	0.56
Tráfico + datos meteorológicos día completo	0.64	0.63	0.63	0.63	0.63
Tráfico en la mañana	0.58	0.58	0.58	0.59	0.57
Tráfico + datos meteorológicos en la mañana	0.68	0.69	0.67	0.68	0.66

El mejor valor del área bajo la curvafue de 0.69 y el peor valor fue de 0.55,se obtuvo con los datos del tráfico en día completo en las diferentes áreas. El valor más bajo del área bajo la curva se obtuvo en Redes Neuronales con los datos del tráfico + datos meteorológicos en un área de 10.9 km^2 como se observa en la Tabla 14.

6. Conclusiones y Recomendaciones

6.1. Conclusiones

Después del estudio realizado el mejor modelo obtenido fue en el tercer método donde se trabajó con capturas de pantalla en un área de 14.0km^2 con los datos del tráfico + datos meteorológicos en horas de la mañana 6am a 10am.

Al evaluar los tres métodos utilizados se deduce que el tratamiento de imágenes de área rectangular es el método más preciso ya que se obtuvo coeficientes de correlación más altos en un rango de 0.4 a 0.65, a diferencia del primer método donde se trabajó con el tráfico representado en tiempo que arrojó un coeficiente de correlación mucho más bajo que es de 0.26.

Los clasificadores se aplicaron solo en el tercer método donde se trabajó con capturas de pantalla en diferentes áreas debido a que en el primer y segundo método no se obtuvieron buenos modelos al aplicar Regresión Lineal por lo cual quedaron descartados.

Es recomendable realizar capturas de pantalla de áreas mayores a 14 km^2 para obtener mejores resultados de coeficientes de correlación, el coeficiente de correlación ideal es de 1, pero con el área de 14 km^2 se obtuvo coeficientes de correlación de 0.52 a 0.65, a diferencia de las otras áreas 2.2, 5.0, 8.7 y 10.9 que se obtuvieron coeficientes de correlación de 0.28 a 0.6.

Al analizar solo datos del tráfico vs PM2.5 se obtuvieron coeficientes de correlación muy bajos como 0.28, a diferencia del análisis realizado del tráfico con datos meteorológicos vs PM2.5 que se obtuvo datos más altos como 0.65.

Los datos analizados en la mañana de 6am a 10am ayudaron a obtener coeficientes de correlación más altos debido a que los valores de los datos meteorológicos que son precipitación, presión, radiación solar, temperatura y viento tienen mayores niveles de concentración en esas horas.

Para obtener mejores resultados al desarrollar el modelo es recomendable utilizar los datos del tráfico sumado los datos meteorológicos.

6.2. Recomendaciones

Es importante revisar los archivos antes de ser cargados en Weka para que no existan datos vacíos ya que Weka no permite subir archivos con datos vacíos.

En base a los coeficientes de correlación obtenidos, para un trabajo futuro se puede trabajar con áreas más extensas, mayores a 14 km^2 para ver si es que el modelo mejora.

Para dar una herramienta que ayude a la ciudadanía o al gobierno de turno a tomar acciones referentes a la contaminación se puede realizar una aplicación basado en los resultados obtenidos en esta tesis.

Referencias

Rybarczyk, Y., y Zalakeviciute, R. (2016). *Machine Learning Approach to Forecasting Urban Pollution*. Quito, Ecuador: Universidad de las Américas

Wikipedia. (s.f.). Inductivismo. Recuperado de https://es.wikipedia.org/wiki/Inductivismo

Gonzalez, M. (2016*). Artificial Intelligence Introduction to Machine Learning*. Quito, Ecuador:CIEDI.

Hauskrecht, M. (2015). *Designing a learning sytem. Pittsburgh, United States of America:CS.*

OMS. (2016). Calidad del aire ambiente (exterior) y salud. Recuperado el 10 de Julio de 2017, de http://www.who.int/mediacentre/factsheets/fs313/es/.

García, J. (2012). *Machine Learning and cognitive systems: the next evolution of enterprise intelligence* (part 1). *Magazine Wired*. Recuperado de https://www.wired.com/insights/2014/07/machine-learning-cognitive-systems-next-evolution-enterprise-intelligence-part/

Wikipedia. (2009). Gráfico de varias curvas ROC. Recuperado de https://es.wikipedia.org/wiki/Curva_ROC#/media/File:Curvas.png

SAS. (s.f). *Machine Learning What it is and why it matters*. Recuperado el 12 de Diciembre de 2017 de https://www.sas.com/en_us/insights/analytics/machine-learning.html

Medium Corporation. Maini, V. Machine Learning for Humans, Part 2.1: *Supervised Learning.* Recuperado el 20 de Diciembre de https://medium.com/machine-learning-for-humans/supervised-learning-740383a2feab

Stat. (s.f.). Chapter 12 Logistic Regression. Recuperado el 20 de Diciembre de http://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch12.pdf

Data Science Central. Sharma, S. Recuperado el 20 de Diciembre de https://www.datasciencecentral.com/profiles/blogs/artificial-neural-network-ann-in-machine-learning

James, G. Witten, D. Hastie, T. Tibshirani, R. (2009). *An Introduction to Statistical Learning with Applications in R. New York, USA: Stringer.*

Witten, I. Frank, E, Hall, M. (2011). *Data Mining Practical Machine Learning Tools and Techniques. Burlington USA: ELSEVIER*

Statsoft. (s.f.). k-Nearest Neighbors. Recuperado el 22 de Diciembre de http://www.statsoft.com/Textbook/k-Nearest-Neighbors#predictions

Ciemat. Cárdenas,M. Recuperado el 23 de Diciembre de http://wwwae.ciemat.es/~cardenas/docs/lessons/KNN.pdf

Svm-tutorial. Kowalczyk, A. (2014). Recuperado el 23 de Diciembre de https://www.svm-tutorial.com/2014/11/svm-understanding-math-part-1/

ANEXOS

Anexo 1. Código fuente para recolección de datos del tráfico desde Google Maps.

```
import requests

import csv

import time

import simplejson

from datetime import datetime

from urllib import FancyURLopener


#-----------------NORTH---------------------------------#


url_north_going = "https://www.google.com.ec/maps/dir/'-0.175222,-
78.500781'/'-0.176553,-78.494258'/@-0.1759544,-
78.4997187,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.500781!2d-
0.175222!1m3!2m2!1d-78.494258!2d-0.176553!3e0"


url_north_return = "https://www.google.com.ec/maps/dir/'-0.176553,-
78.494258'/'-0.175222,-78.500781'/@-0.1759544,-
78.4997187,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.494258!2d-
0.176553!1m3!2m2!1d-78.500781!2d-0.175222!3e0"


#-----------------WEST---------------------------------#


url_west_going = "https://www.google.com.ec/maps/dir/'-0.175893,-
78.501690'/'-0.179884,-78.504264'/@-0.1779683,-
78.5052676,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.50169!2d-
0.175893!1m3!2m2!1d-78.504264!2d-0.179884!3e0"


url_west_return = "https://www.google.com.ec/maps/dir/'-0.179910,-
78.504141'/-0.176080,+-78.501620/@-0.1778669,-
78.5049957,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.504141!2d-
0.17991!1m3!2m2!1d-78.50162!2d-0.17608!3e0"
```

```
#-----------------SOUTH---------------------------------#


url_south_going = "https://www.google.com.ec/maps/dir/+-0.182097,-
78.504530/'-0.185868,-78.497576'/@-0.1839912,-
78.5032545,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.50453!2d-
0.182097!1m3!2m2!1d-78.497576!2d-0.185868!3e0"


url_south_return = "https://www.google.com.ec/maps/dir/'-0.185675,-
78.497490'/'-0.181984,-78.504517'/@-0.1837959,-
78.5032211,17z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.49749!2d-
0.185675!1m3!2m2!1d-78.504517!2d-0.181984!3e0"


#-----------------EAST---------------------------------#


url_east_going = "https://www.google.com.ec/maps/dir/'-0.177459,-78.494225'/'-
0.184132,-78.495276'/@-0.1810127,-
78.4990065,16z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.494225!2d-
0.177459!1m3!2m2!1d-78.495276!2d-0.184132!3e0"


url_east_return = "https://www.google.com.ec/maps/dir/'-0.184991,-
78.495193'/+-0.177459,-78.493970/@-0.1809778,-
78.4989024,16z/data=!3m1!4b1!4m10!4m9!1m3!2m2!1d-78.495193!2d-
0.184991!1m3!2m2!1d-78.49397!2d-0.177459!3e0"


def escribir():

    m1=open("prueba_fin1.csv","at")

    m1_csv=csv.writer(m1)

    #columna1=[["Date-Time","North","West","South","West"]]

    columna1=[["Date-Time","N-WE time","N-EW time","W-NS time", "W-SN
time","S-WE time","S-EW time","E-NS time", "E-SN time"]]

    m1_csv.writerows(columna1)

    m1.close()
```

```
escribir()



while True:


    #-----------------NORTH----------------------------------#
    class MyOpener(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'


    myopener = MyOpener()
    pag = myopener.open(url_north_going)


    pag1 = pag.read()
    posm1 = pag1.find(' min\\"]')


    posm2 = pag1[posm1+4:].find(' min\\"]')


    duration_going_n = pag1[posm1 + posm2 + 2: posm1 + posm2 + 4]
    #print duration_going_n
    duration_going_n1 = pag1[posm1 + posm2 + 3: posm1 + posm2 + 4]
    #print duration_going_n1


    if duration_going_n[0] == "":
       print duration_going_n1
       n_going = pag1[posm1 + posm2 + 3: posm1 + posm2 + 4]
    else:
        print duration_going_n
        n_going = pag1[posm1 + posm2 + 2: posm1 + posm2 + 4]
```

```
    n_going1 = float(n_going)
    print n_going1


    class MyOpener1(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'


    myopener1 = MyOpener1()
    pag2 = myopener1.open(url_north_return)
    pag3 = pag2.read()


    posm3 = pag3.find(' min\\"]')
    posm4 = pag3[posm3+4:].find(' min\\"]')
    duration_return_n = pag3[posm3 + posm4 + 2: posm3 + posm4 + 4]
    duration_return_n1 = pag3[posm3 + posm4 + 3: posm3 + posm4 + 4]
    if duration_return_n[0] == "":
       print duration_return_n1
       n_return = pag3[posm3 + posm4 + 3: posm3 + posm4 + 4]
    else:
        print duration_return_n
        n_return = pag3[posm3 + posm4 + 2: posm3 + posm4 + 4]
    n_return1 = float(n_return)




    #----------------WEST----------------------------------#


    class MyOpener2(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'
```

```
    myopener2 = MyOpener2()
    pag4 = myopener2.open(url_west_going)
    pag5 = pag4.read()

    posm5 = pag5.find(' min\\"]')
    posm6 = pag5[posm5+4:].find(' min\\"]')

    duration_going_w = pag5[posm5 + posm6 + 2: posm5 + posm6 + 4]
    #print duration_going_w
    duration_going_w1 = pag5[posm5 + posm6 + 3: posm5 + posm6 + 4]
    #print duration_going_w1

    if duration_going_n[0] == "":
       print duration_going_w1
       w_going = pag5[posm5 + posm6 + 3: posm5 + posm6 + 4]
    else:
        print duration_going_w
        w_going = pag5[posm5 + posm6 + 2: posm5 + posm6 + 4]
    w_going1 = float(w_going)



    class MyOpener3(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'

    myopener3 = MyOpener3()
    pag6 = myopener3.open(url_west_return)
    pag7 = pag6.read()
```

```
    posm7 = pag7.find(' min\\"]')
    posm8 = pag7[posm7+4:].find(' min\\"]')

    duration_return_w = pag7[posm7 + posm8 + 2: posm7 + posm8 + 4]
    duration_return_w1 = pag7[posm7 + posm8 + 3: posm7 + posm8 + 4]
    if duration_return_w[0] == "":
       print duration_return_w1
       w_return = pag7[posm7 + posm8 + 3: posm7 + posm8 + 4]
    else:
        print duration_return_w
        w_return = pag7[posm7 + posm8 + 2: posm7 + posm8 + 4]
    w_return1 = float(w_return)




    #-----------------SOUTH-----------------------------------#

    class MyOpener4(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'

    myopener4 = MyOpener4()
    pag8 = myopener4.open(url_south_going)
    pag9 = pag8.read()

    posm9 = pag9.find(' min\\"]')
    posm10 = pag9[posm9+4:].find(' min\\"]')
```

```
    duration_going_s = pag9[posm9 + posm10 + 2: posm9 + posm10 + 4]
    duration_going_s1 = pag9[posm9 + posm10 + 3: posm9 + posm10 + 4]

    if duration_going_s[0] == "":
       print duration_going_s1
       s_going = pag9[posm9 + posm10 + 3: posm9 + posm10 + 4]
    else:
        print duration_going_s
        s_going = pag9[posm9 + posm10 + 2: posm9 + posm10 + 4]
    s_going1 = float(s_going)



    class MyOpener5(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'


    myopener5 = MyOpener5()
    pag10 = myopener5.open(url_south_return)
    pag11 = pag10.read()

    posm11 = pag11.find(' min\\"]')
    posm12 = pag11[posm11+4:].find(' min\\"]')

    duration_return_s = pag11[posm11 + posm12 + 2: posm11 + posm12 + 4]
    duration_return_s1 = pag11[posm11 + posm12 + 3: posm11 + posm12 + 4]

    if duration_return_s[0] == "":
       print duration_return_s1
       s_return = pag11[posm11 + posm12 + 3: posm11 + posm12 + 4]
```

```
    else:
        print duration_return_s
        s_return = pag11[posm11 + posm12 + 2: posm11 + posm12 + 4]
    s_return1 = float(s_return)

    #----------------EAST---------------------------------#

    class MyOpener6(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'

    myopener6 = MyOpener6()
    pag12 = myopener6.open(url_east_going)
    pag13 = pag12.read()

    posm13 = pag13.find(' min\\"]')
    posm14 = pag13[posm13+4:].find(' min\\"]')

    duration_going_e = pag13[posm13 + posm14 + 2: posm13 + posm14 + 4]
    duration_going_e1 = pag13[posm13 + posm14 + 3: posm13 + posm14 + 4]

    if duration_going_e[0] == "":
       print duration_going_e1
       e_going = pag13[posm13 + posm14 + 3: posm13 + posm14 + 4]
    else:
        print duration_going_e
        e_going = pag13[posm13 + posm14 + 2: posm13 + posm14 + 4]
    e_going1 = float(e_going)
```

```
    class MyOpener7(FancyURLopener):
        version = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36'

    myopener7 = MyOpener7()
    pag14 = myopener7.open(url_east_return)
    pag15 = pag14.read()

    posm14 = pag15.find(' min\\"]')
    posm15 = pag15[posm14+4:].find(' min\\"]')

    duration_return_e = pag15[posm14 + posm15 + 2: posm14 + posm15 + 4]
    duration_return_e1 = pag15[posm14 + posm15 + 3: posm14 + posm15 + 4]

    if duration_return_e[0] == "":
       print duration_return_e1
       e_return = pag15[posm14 + posm15 + 3: posm14 + posm15 + 4]
    else:
        print duration_return_e
        e_return = pag15[posm14 + posm15 + 2: posm14 + posm15 + 4]
    e_return1 = float(e_return)

    time1 = str(datetime.now())
    time2 = time1[0:19]
    m2=open("prueba_fin1.csv","at")
    m2_csv=csv.writer(m2)

re=[[time2,n_going1,n_return1,w_going1,w_return1,s_going1,s_return1,e_going
1,e_return1]]
    #re=[[time2,n_going1]]
```

```
    m2_csv.writerows(re)
    m2.close()
    time.sleep(600)
```

Anexo 2. Código fuente para realizar capturas de pantalla del tráfico de Google Maps.

```
from selenium import webdriver
import time
import csv
from datetime import datetime


def escribir(): #método para inicializar el archivo
    m1=open("registro_imagenes.csv","at")
    m1_csv=csv.writer(m1)
    m1.close()
escribir()


driver = webdriver.Firefox()
#driver = webdriver.Firefox(executable_path=r'C:\Program Files (x86)\Mozilla Firefox\firefox.exe',capabilities={"marionette":False})
#En caso de que no encuentra el path de firefox o tenga instalado otra versión de firefox y selenium


driver.get("https://www.google.com/maps/@-0.181661,-78.4987077,15.5z/")# Guardado de imagen estática 1
driver.save_screenshot("picturea_0.jpg")
imagea = "picturea_0.jpg"
driver.get("https://www.google.com/maps/@-0.1824979,-78.4995444,15z/")# Guardado de imagen estática 2
driver.save_screenshot("pictureb_0.jpg")
imageb = "pictureb_0.jpg"
```

```
driver.get("https://www.google.com/maps/@-0.1832142,-78.4978562,14.5z/")#
Guardado de imagen estática 3

driver.save_screenshot("picturec_0.jpg")

imagec = "picturec_0.jpg"

driver.get("https://www.google.com/maps/@-0.1809014,-78.4964736,14z/")#
Guardado de imagen estática 4

driver.save_screenshot("pictured_0.jpg")

imaged = "pictured_0.jpg"

driver.get("https://www.google.com/maps/@-0.1792707,-78.4898646,13z/")#
Guardado de imagen estática 5

driver.save_screenshot("picturee_0.jpg")

imagee = "picturee_0.jpg"


def escribir(): #método para guardar las imágenes estáticas
    m1=open("registro_imagenes.csv","at")
    m1_csv=csv.writer(m1,lineterminator='\n')
    columna1 = [[imagea]]
    columna2 = [[imageb]]
    columna3 = [[imagec]]
    columna4 = [[imaged]]
    columna5 = [[imagee]]
    m1_csv.writerows(columna1)
    m1_csv.writerows(columna2)
    m1_csv.writerows(columna3)
    m1_csv.writerows(columna4)
    m1_csv.writerows(columna5)
    m1.close()
escribir()


i = 1
```

```
while True:
    try:
        driver.get("https://www.google.com/maps/@-0.181661,-
78.4987077,15.5z/data=!5m1!1e1")# Guardado de imagen continua 1
        driver.save_screenshot("picturea_" + str(i) + ".jpg")
        picturea = "picturea_" + str(i)
        print(picturea)

        driver.get("https://www.google.com/maps/@-0.1824979,-
78.4995444,15z/data=!5m1!1e1")# Guardado de imagen continua 2
        driver.save_screenshot("pictureb_" + str(i) + ".jpg")
        pictureb = "pictureb_" + str(i)
        print(pictureb)

        driver.get("https://www.google.com/maps/@-0.1832142,-
78.4978562,14.5z/data=!5m1!1e1")# Guardado de imagen continua 3
        driver.save_screenshot("picturec_" + str(i) + ".jpg")
        picturec = "picturec_" + str(i)
        print(picturec)

        driver.get("https://www.google.com/maps/@-0.1809014,-
78.4964736,14z/data=!5m1!1e1")# Guardado de imagen continua 4
        driver.save_screenshot("pictured_" + str(i) + ".jpg")
        pictured = "pictured_" + str(i)
        print(pictured)

        driver.get("https://www.google.com/maps/@-0.1792707,-
78.4898646,13z/data=!5m1!1e1")# Guardado de imagen continua 5
        driver.save_screenshot("picturee_" + str(i) + ".jpg")
        picturee = "picturee_" + str(i)
```

```
        print(picturee)


        i = i + 1


        time1 = str(datetime.now())
        time2 = time1[0:19]
        m2=open("registro_imagenes.csv","at")
        m2_csv=csv.writer(m2,lineterminator='\n')
        re1=[[picturea,time2]]
        re2=[[pictureb,time2]]
        re3=[[picturec,time2]]
        re4=[[pictured,time2]]
        re5=[[picturee,time2]]
        m2_csv.writerows(re1)
        m2_csv.writerows(re2)
        m2_csv.writerows(re3)
        m2_csv.writerows(re4)
        m2_csv.writerows(re5)
        m2.close()
    except Exception:
        print("No internet!!!")
        pass


    time.sleep(600)
driver.close()
```

Anexo 3. Código fuente para realizar la extracción de colores de la captura de pantalla del tráfico de Google Maps.

```
// The overall data must be copied in the file: pixelColors.xlsx


PrintWriter output;
int QFrame = 1400; // by 500 steps of pictures, because of the limitation of memory
PImage[] frame = new PImage[QFrame];


void setup() {


  output = createWriter("pixelColors10.csv");
  size(500,500);


  for(int i = 1353; i < QFrame; i++) {
    frame[i] = loadImage("picture_" + (i+1) + "_crop" + ".jpg");
  }


  for(int i = 1353; i < QFrame; i++) {
    float rouge = 0;
    float orange = 0;
    float vert = 0;
    image(frame[i], 0, 0);
    for(int x=0; x<width; x++) {
      for(int y=0; y<height; y++) {
        color c = get(x, y);
        if(red(c) < 200 || green(c) < 200 || blue(c) < 200) { // to remove white pixels
          if(red(c) > green(c) && abs(green(c)-blue(c)) < 20) {
            rouge++;
          } else if(red(c) > green(c) && green(c) > blue(c)) {
```

```
          orange++;
        } else if(green(c) > red(c) && red(c) > blue(c)) {
          vert++;
        }
      }
    }
   }
   output.print("picture_" + (i+1) + ",");
   output.print(rouge/(rouge+orange+vert) + ","); // % red per picture
   output.print(orange/(rouge+orange+vert) + ","); // % orange per picture
   output.print(vert/(rouge+orange+vert) + ","); // % vert per picture
   output.println();
  }

  output.flush();
  output.close();
  exit();
}
```

Anexo 4. Código fuente para graficar las curva ROC.

```
import numpy as np
from scipy import interp
import matplotlib.pyplot as plt
from itertools import cycle
import pandas as pd

from sklearn import svm, datasets
from sklearn.metrics import roc_curve, auc
```

```
from sklearn.model_selection import StratifiedKFold

from sklearn.linear_model import LogisticRegression

from sklearn import neighbors

from sklearn.neural_network import MLPClassifier


#
##############################################################################
###############

# Data IO and generation


# Import some data to play with

def data(filename):

    X = pd.read_table(filename, sep=',', warn_bad_lines=True,
error_bad_lines=True, low_memory = False)


    X = np.asarray(X)


    data = X[:,1:]

    print data

    labels = X[:,10]


    print np.unique(labels)


    return data, labels


filename = 'D:/JESSIE/tesis 05122017/clasificadores/j48/crop
5/mañana/crop_decimo_trafico_meteorologia_mañana_J48.csv'

X, y = data(filename)

print filename

n_samples, n_features = X.shape
```

```
# Add noisy features
random_state = np.random.RandomState(0)
X = np.c_[X, random_state.randn(n_samples, 200 * n_features)]

#
##############################################################################
# Classification and ROC analysis

# Run classifier with cross-validation and plot ROC curves
cv = StratifiedKFold(n_splits=10)

#SVM
classifier = svm.SVC(kernel='linear', probability=True,
                     random_state=random_state)

tprs = []
aucs = []
mean_fpr = np.linspace(0, 1, 100)

i = 0
for train, test in cv.split(X, y):
    probas_ = classifier.fit(X[train], y[train]).predict_proba(X[test])
    # Compute ROC curve and area the curve
    fpr, tpr, thresholds = roc_curve(y[test], probas_[:, 1])
    tprs.append(interp(mean_fpr, fpr, tpr))
    tprs[-1][0] = 0.0
    roc_auc = auc(fpr, tpr)
    aucs.append(roc_auc)
```

```
#    plt.plot(fpr, tpr, lw=1, alpha=0.3,           ESTA PARTE ES DE LOS FOLD IMPRESOS
#             label='ROC fold %d (AUC = %0.2f)' % (i, roc_auc))

    i += 1
plt.plot([0, 1], [0, 1], linestyle='--', lw=2, color='r',
         label='', alpha=.8)

mean_tpr = np.mean(tprs, axis=0)
mean_tpr[-1] = 1.0
mean_auc = auc(mean_fpr, mean_tpr)
std_auc = np.std(aucs)
plt.plot(mean_fpr, mean_tpr, color='b',
         label=r'SVM (AUC = %0.2f $\pm$ %0.2f)' % (mean_auc, std_auc),
         lw=2, alpha=.8)

#logistic regression

classifier1 = LogisticRegression()
classifier1.fit(X, y)

tprs1 = []
aucs1 = []
mean_fpr1 = np.linspace(0, 1, 100)

i = 0
for train, test in cv.split(X, y):
    probas_ = classifier1.fit(X[train], y[train]).predict_proba(X[test])
    # Compute ROC curve and area the curve
```

```
    fpr1, tpr1, thresholds1 = roc_curve(y[test], probas_[:, 1])
    tprs1.append(interp(mean_fpr1, fpr1, tpr1))
    tprs1[-1][0] = 0.0
    roc_auc1 = auc(fpr1, tpr1)
    aucs1.append(roc_auc1)

    i += 1

mean_tpr1 = np.mean(tprs1, axis=0)
mean_tpr1[-1] = 1.0
mean_auc1 = auc(mean_fpr1, mean_tpr1)
std_auc1 = np.std(aucs1)
plt.plot(mean_fpr1, mean_tpr1, color='green',
         label=r'Logistic Regression (AUC = %0.2f $\pm$ %0.2f)' % (mean_auc1,
std_auc1),
         lw=2, alpha=.8)

#knn

classifier2 = neighbors.KNeighborsClassifier(n_neighbors=15)
classifier2.fit(X, y)

tprs2 = []
aucs2 = []
mean_fpr2 = np.linspace(0, 1, 100)

i = 0
for train, test in cv.split(X, y):
    probas_ = classifier2.fit(X[train], y[train]).predict_proba(X[test])
```

```
    # Compute ROC curve and area the curve
    fpr2, tpr2, thresholds2 = roc_curve(y[test], probas_[:, 1])
    tprs2.append(interp(mean_fpr2, fpr2, tpr2))
    tprs2[-1][0] = 0.0
    roc_auc2 = auc(fpr2, tpr2)
    aucs2.append(roc_auc2)

    i += 1

mean_tpr2 = np.mean(tprs2, axis=0)
mean_tpr2[-1] = 1.0
mean_auc2 = auc(mean_fpr2, mean_tpr2)
std_auc2 = np.std(aucs2)
plt.plot(mean_fpr2, mean_tpr2, color='purple',
        label=r'KNN (AUC = %0.2f $\pm$ %0.2f)' % (mean_auc2, std_auc2),
        lw=2, alpha=.8)

#neural network

classifier3 = MLPClassifier(solver='lbfgs', alpha=1e-5,
                    hidden_layer_sizes=(5, 2), random_state=random_state)
classifier3.fit(X, y)

tprs3 = []
aucs3 = []
mean_fpr3 = np.linspace(0, 1, 100)

i = 0
for train, test in cv.split(X, y):
```

```
    probas_ = classifier3.fit(X[train], y[train]).predict_proba(X[test])
    # Compute ROC curve and area the curve
    fpr3, tpr3, thresholds3 = roc_curve(y[test], probas_[:, 1])
    tprs3.append(interp(mean_fpr3, fpr3, tpr3))
    tprs3[-1][0] = 0.0
    roc_auc3 = auc(fpr3, tpr3)
    aucs3.append(roc_auc3)

    i += 1

mean_tpr3 = np.mean(tprs3, axis=0)
mean_tpr3[-1] = 1.0
mean_auc3 = auc(mean_fpr3, mean_tpr3)
std_auc3 = np.std(aucs3)
plt.plot(mean_fpr3, mean_tpr3, color='cyan',
         label=r'Neural Network (AUC = %0.2f $\pm$ %0.2f)' % (mean_auc3,
std_auc3),
         lw=2, alpha=.8)


plt.xlim([-0.05, 1.05])
      plt.ylim([-0.05, 1.05])
      plt.xlabel('Especificidad')
      plt.ylabel('Sensibilidad')
      plt.title('Curva ROC')
      plt.legend(loc="lower right")
      plt.show()
```