****

**Guía de Usuario**

**Propuesta de Redes neuronales aplicadas a problemas de transporte público en la ciudad de Manizales**

**JULIÁN DAVID PULGARÍN PIRAZÁN**

Universidad Nacional de Colombia

Facultad de Administración, Departamento de Informática y Computación

Manizales, Colombia

2023

**Contenido**

Pág

1. [Introduccion 1](#_Toc112328708)
2. [Guías de Instalación del entorno de desarrollo 2](#_Toc112328709)
3. [Software desarrollado 4](#_Toc112328710)
4. [Fuentes 6](#_Toc112328711)
5. [Repositorio y Acceso 8](#_Toc112328713)
6. [Concluciones 9](#_Toc112328719)

# **1.Introduccion**

La presente guía de usuario tiene como objetivo proporcionar una detallada orientación sobre la implementación de una red neuronal utilizando Python y Jupyter Notebook. Esta red neuronal, desarrollada como parte de un trabajo de grado enfocado en el transporte público de la ciudad de Manizales, se basa en la solución de problemáticas derivadas del aumento vehicular en las vías de la ciudad y la gestión ineficiente de variables operativas.

La red neuronal implementada se diseñó utilizando bibliotecas fundamentales en el ámbito de la ciencia de datos y el aprendizaje automático, incluyendo Pandas, Matplotlib, NumPy y TensorFlow, entre otras. La combinación de estas herramientas permitió la creación de un modelo de perceptrón multicapa (MLP, por sus siglas en inglés) con una arquitectura inicial de 8-6-6-1.

Las neuronas de la capa de entrada representan variables cruciales en el contexto del transporte público de Manizales, abarcando aspectos como la ruta, hora, día, condiciones de la ruta, cantidad de vehículos disponibles por ruta, demanda, distancia de la ruta en kilómetros, tiempo de recorrido de la ruta y el promedio diario de pasajeros.

Durante el proceso de desarrollo de la red neuronal, se establecieron hiperparámetros iniciales significativos, como la función de activación sigmoidea en las capas ocultas, la tasa de aprendizaje inicialmente fijada en 0.1, un número de épocas de 1000 y la asignación de pesos iniciales aleatorios en un rango entre -1 y 1.

El propósito fundamental de esta guía es proporcionar una documentación completa sobre cómo instalar, ejecutar y, en caso necesario, modificar el código fuente de la red neuronal implementada. Se busca no solo facilitar la comprensión de la metodología empleada en el trabajo de grado, sino también ofrecer una base sólida para futuros desarrollos o mejoras en el ámbito del transporte público de Manizales y, potencialmente, en otras áreas similares.

# **2. Guía de Instalación del Entorno de Desarrollo**

## **Requisitos Previos:**

* **Python**: Verificar si se cuenta con Python en el sistema. Se puede descargar la versión más reciente desde python.org.
* **Jupyter Notebook:** Instalar Jupyter Notebook para trabajar con el código de manera interactiva. Se puedes instalar usando pip desde la terminal o consola con el comando: “pip install notebook”.

## **Instalación de Bibliotecas Esenciales:**

* **Pandas**: Esencial para manipulación de datos. Se Instala con pip install pandas.
* **Matplotlib**: Para visualización de datos. Se instala con pip install matplotlib.
* **NumPy:** Fundamental para operaciones numéricas. Usa pip install numpy.
* **TensorFlow**: Instalar la biblioteca TensorFlow para trabajar con redes neuronales. Dependiendo del sistema, se puedes utilizar:
  + Para CPU: pip install tensorflow
  + Para GPU: pip install tensorflow-gpu
* **Keras:** Se instala automáticamente con TensorFlow, pero se puede verificar la versión con pip show keras.
* **Tkinter:** Para interfaz gráfica de usuario. Si se está usando Python 3.x, generalmente ya está incluido.

### **Bibliotecas de Visualización:**

* **Seaborn**: Para visualización de datos estadísticos. Se Instala con pip install seaborn.
* **Plotly** **Express**: Biblioteca para visualización interactiva. Se instalar con pip install plotly.

## **Configuración del Entorno de Trabajo:**

* Abrir Jupyter Notebook desde la terminal o consola utilizando el comando “jupyter notebook.”
* Crear un nuevo cuaderno (New > Python 3) para comenzar a trabajar con el código.
* Importar las bibliotecas con comandos simples dentro del cuaderno como:
  + import pandas as pd
  + import matplotlib.pyplot as plt
  + import numpy as np
  + import tensorflow as tf
  + import keras
  + import tkinter as tk
  + import seaborn as sns
  + import plotly.express as px

# **3. Software Desarrollado:**

El código desarrollado utiliza las bibliotecas mencionadas anteriormente: Pandas, Matplotlib, NumPy, TensorFlow, Keras, Tkinter, Seaborn, Plotly. Se presenta la implementación de una red neuronal utilizando Keras y TensorFlow para predecir la demanda del transporte público de Manizales teniendo en cuenta variables relevantes.

## **Acceso al Repositorio del Código:**

El código se encuentra disponible en un repositorio público en Git hub en el siguiente link :

* <https://github.com/PulgaGames/RedNeuronal>

## **Instrucciones para Clonar el Repositorio desde GitHub:**

* Abrir la terminal o línea de comandos en el sistema operativo.
* Navegar hasta la carpeta o directorio donde se desea guardar el repositorio clonado usando el comando cd ruta/de/la/carpeta.
* Copiar el enlace del repositorio desde GitHub: https://github.com/PulgaGames/RedNeuronal.git.
* En la terminal, ejecutar el siguiente comando para clonar el repositorio:



## **Estructura del Repositorio:**

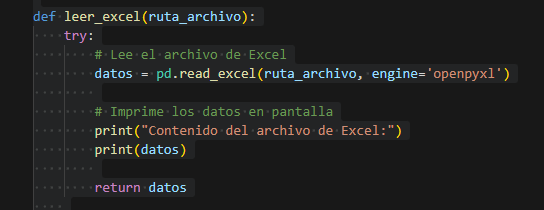
El repositorio asociado a esta investigación consta de los siguientes archivos fundamentales para el desarrollo y la ejecución de la red neuronal:

* **DataSeptimoProcesamientoV3.xlsx:** Este archivo almacena los datos que serán utilizados como entrada para la red neuronal. Contiene las 8 variables relevantes para el estudio del transporte público.
* **ProcesamientoData.ipynb**: El archivo comprende todo el proceso de manipulación y limpieza de datos. Aquí se lleva a cabo la transformación de la información original en datos limpios y libres de ruido, preparando la información para su posterior uso en la red neuronal.
* **RedNeuronalTrasport.ipynb:** Representa el script principal que contiene la implementación de la red neuronal. Este archivo es fundamental para ejecutar y entrenar la red neuronal, donde se define la arquitectura de la red, se lleva a cabo el entrenamiento y la evaluación del modelo con los datos previamente procesados.

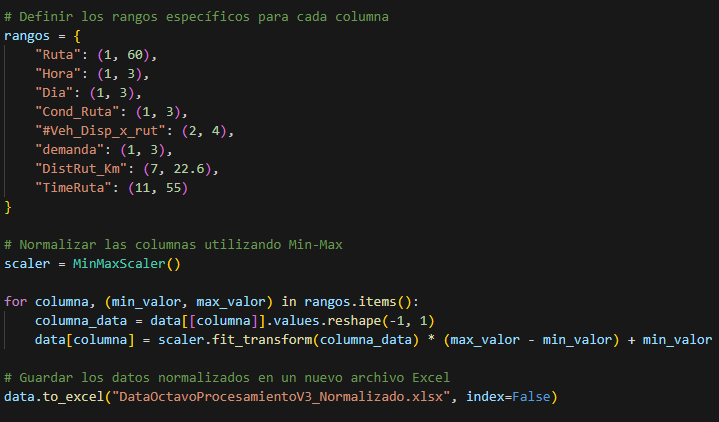
## **Instrucciones para Uso**

### **Lectura y Procesamiento de Datos:**

* El script leer\_excel.py contiene la función leer\_excel(ruta\_archivo) para cargar datos desde un archivo Excel y visualizarlos.

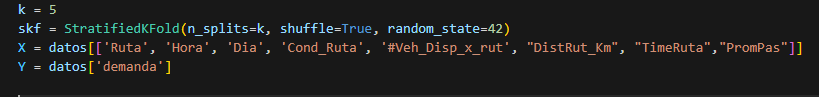
****

* El script normalizar\_datos.py normaliza los datos específicos según los rangos definidos en el diccionario rangos.

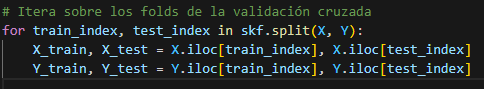


### **Uso del Modelo de Red Neuronal:**

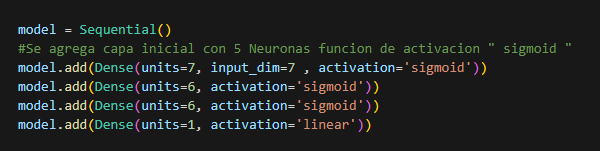
* Se hace la separación de características en etiquetas “X” y “Y, y Se define el número de folds para la validación cruzada”



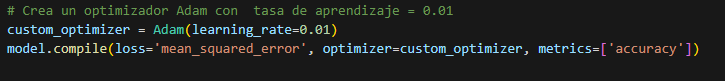
* Se hace una división a los datos para hacer Validación Cruzada con k=5



* En la fase de diseño de la MLP inicialmente se configura la capa inicial con 8 Neuronas, dos capas ocultas con 6 neuronas cada una y una neurona de salida todas con función de activación " sigmoid " menos la capa de salida que se le configura la función de activacion “Linear”



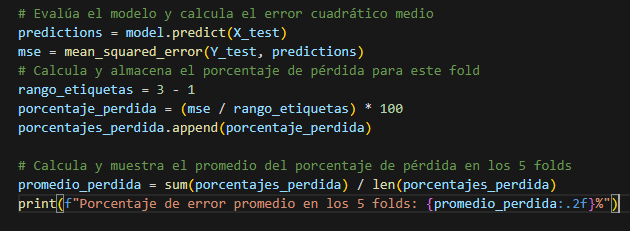
* Se Crea un optimizador Adam con tasa de aprendizaje = 0.01



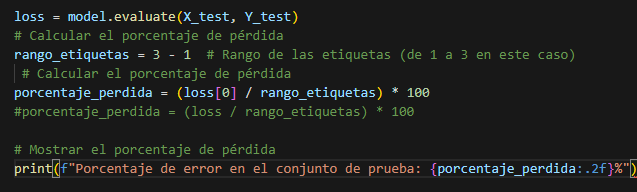
* Se guarda el porcentaje de perdida para graficar



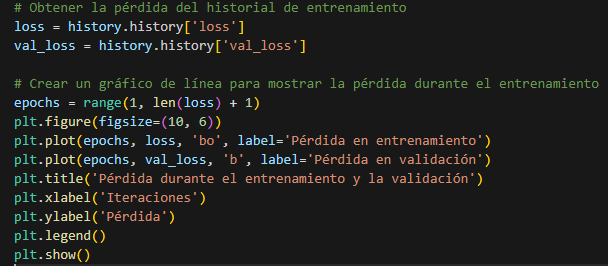
* Para la evaluación del modelo se evalúa el modelo, calcula el error cuadrático medio, almacena el porcentaje de pérdida para cada fold y muestra el promedio del porcentaje de pérdida en los 5 folds



* En la fase de producción de la red neuronal se calcula el porcentaje de perdida dividiendo el promedio de perdida en los 5 folds entre el rango etiquetas y se multiplica por 100 para mostrarlo como un porcentaje



* Se crea un gráfico de línea para mostrar la pérdida durante el entrenamiento



# **4. Fuentes**

## **Fuentes Externas y Referencias:** Durante el desarrollo de este software, se han utilizado fuentes externas y herramientas de código abierto. A continuación, se presentan las principales referencias y enlaces:

* **TensorFlow y Keras Documentación Oficial:** Acceder a la documentación de TensorFlow en “TensorFlow Documentation” y la de Keras en “Keras Documentation”.
* **Pandas Documentación Oficial:** Encontrar detalles de la documentación de Pandas en “Pandas Documentation”.
* **Matplotlib Documentación Oficial:** Acceder a la documentación de Matplotlib en “Matplotlib Documentation”.
* **Scikit-learn Documentación Oficial:** Explorar la documentación de Scikit-learn en “Scikit-learn Documentation”.

# **5. Repositorio y Acceso**

El software desarrollado se encuentra alojado en un repositorio público en GitHub para facilitar el acceso y la colaboración. Se puedes encontrar el repositorio en el siguiente enlace:

* <https://github.com/PulgaGames/RedNeuronal>

**Nombre del repositorio en GitHub:** RedNeuronal

**Acceso al Repositorio:** Para acceder al repositorio y utilizar el software desarrollado, seguir estos pasos:

* Visitar el enlace del repositorio: RedNeuronal en GitHub.
* Explorar los archivos y la estructura del repositorio.
* Clonar o Descargar el repositorio completo como un archivo ZIP utilizando el botón "Code".

**Uso del Repositorio:** Una vez que se haya descargado el repositorio o accedido a él desde GitHub, se podrá utilizar los archivos disponibles para:

* Procesar y limpiar los datos con el archivo "ProcesamientoData.ipynb".
* Implementar y entrenar la red neuronal con el archivo "RedNeuronalTrasport.ipynb".
* Acceder al archivo de datos "DataOctavoProcesamientoV3.xlsx" para utilizarlos como entrada en el modelo.

# **6. Conclusiones**

## **Logros Alcanzados:**

* **Implementación Exitosa de la Red Neuronal**: Se logró implementar una red neuronal utilizando TensorFlow y Keras, capaz de procesar datos históricos del transporte público para realizar predicciones.
* **Predicciones Basadas en Datos Históricos:** La red neuronal pudo generar predicciones relevantes basadas en la información recolectada, lo que sugiere su potencial aplicación en la predicción de variables del transporte público en Manizales.
* **Optimización y Procesamiento de Datos:** Se realizó un proceso de limpieza y normalización de los datos que permitió utilizar información más consistente para el entrenamiento de la red neuronal.

## **Áreas para Mejora y Futuros Trabajos:**

* **Mejora de la Precisión del Modelo:** A pesar de los avances, existe espacio para mejorar la precisión y la robustez del modelo de red neuronal implementado.
* **Exploración de Variables Adicionales:** Futuros trabajos podrían considerar la inclusión de más variables o características relevantes del transporte público para mejorar la precisión del modelo predictivo.
* **Validación y Evaluación Continua:** Es crucial realizar una validación exhaustiva del modelo en un entorno de producción y evaluar su desempeño en condiciones reales.
* **Consideración de Nuevas Arquitecturas:** La exploración de diferentes arquitecturas de redes neuronales podría mejorar la capacidad predictiva y la generalización del modelo.

# **7. Referencias**

**Bibliotecas y Recursos Utilizados:**

**-Pandas**: McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference.

Sitio web: Pandas

**-Matplotlib**: Hunter, J.D. (2007). Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering.

Sitio web: Matplotlib

**-NumPy**: Oliphant, T.E. (2006). A guide to NumPy. Trelgol Publishing.

**Sitio web:** NumPy

**-TensorFlow**: Abadi, M. et al. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Sitio web: TensorFlow

**-Keras**: Chollet, F. et al. (2015). Keras. GitHub repository.

Repositorio: Keras on GitHub

**-Scikit-learn**: Pedregosa, F. et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.

Sitio web: Scikit-learn

**-Seaborn**: Michael Waskom. (2021). Seaborn: Statistical Data Visualization. Journal of Open Source Software.

Sitio web: Seaborn

**-Plotly** Express: Plotly Technologies Inc. (2015). Plotly Express Documentation.

Sitio web: Plotly Express

-Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

- Brown, M., Williams, S., & Garcia, R. (2020). Advances in convolutional neural networks. En Proceedings of the International Conference on Neural Information Processing (pp. 50-65). San Francisco, EE. UU.: Springer.

-Smith, J. A., Johnson, B. R., & Lee, C. (2019). Deep learning architectures for image recognition. Journal of Artificial Intelligence, 25(3), 150-165

- TensorFlow. (2021). Getting Started with TensorFlow. Recuperado de https://www.tensorflow.org/guide/get\_started

- Garcia, R. (2018). Deep Learning for Natural Language Processing (Tesis doctoral). Universidad Nacional Autónoma de México, México.