Consideraciones sobre el proyecto de uso de framework para el aprendizaje máquina

Requerimientos

El proyecto necesita de las librerias de pandas, numpy, matplotlib, dataprep, sklearn, tensorflow, y tensorflow_docs.

Recursos

El proyecto hace uso de set de datos obtenidos mediante el codigo de [implementacion de aprendizaje maquina sin framework](./Aprendizaje_maquina.py), sin embargo no es necesario correro dicho codigo para corre el [proyecto con implementacion de framework](./framework.ipynb).

Reporte sobre implementacion de framework para el aprendizaje máquina

Resumen

En este proyecto se presenta las implementaciones de diversos modelos de regresion lineal desarrollados en tensorflow, con el fin de predecir el peso en gramos de los peces en venta de un mercado en base a sus caracteristicas y especie.

ETL

La etapa de ETL se realizo en la [implementacion de aprendizaje maquina sin framework](./Aprendizaje_maquina.py), y decidi reutilizar el resultado de esta etapa al tratarse del mismo set de datos.

Extract

Se realizo una extraccion total del set de datos Fish.csv de [kaggle](https://www.kaggle.com/datasets/aungpyaeap/fish-market) conformado por:

- Species (variable categorica)
- Weight (peso en gramos)
- Length1 (longitud vertical en cm)
- Length2 (longitud diagonal en cm)
- Length3 (longitud cruzada en cm)
- Height (en cm)
- Width (en cm)

Se utilizo dataprep para generar un reporte del datase, del cual encontr la correlacion entre las columnas Length2 y Length3, por lo cual fueron eliminadas

Tranform

Se realizo una sustitucion de valores mediante one hot enconding con la libreria de sklearn para la columna de Species.

Load

El resultado final de la etapa de ETL fue exportado a clean_fish.csv

Dataset split

Mediante la libreria sklearn se realizo una division de los datos presentes en clean_fish.csv en una proporcion de 66/33, en donde el 66% de los datos estan orientados al entrenamiento y el 33% para la las

pruebas. En los modelos se utilizo el 20% de los datos de entrada con el fin de validacion, por lo que al final nos queda una proporcion de 52.8/13.2/33, en donde el 52.8% de los datos son para el entrenamiento, 13.2% para la evaluacion, y 33% para las pruebas.

Modelos

Todos los modelos fueron entrenados mediante el algoritmo de optimizacion Adam, con una tasa de aprendizaje de 0.001, midiendo el error mediante el promedio del error absoluto, durante 100 epocas. En cada modelo se utilizo una validacion del 20% de los datos de entrada.

Arquitecturas

- Regresion lineal:

  Se implemento mediante una unica capa de una neurona.

- Red neuronal diminuta

  Se implmento una red neuronal de dos capas, con 64 neronas y activacion mediante relu para la primera capa, y una unica nerona para la segunda.

- Red neuronal pequeña

  Se implmento una red neuronal de tres capas, con 64 neronas y activacion mediante relu para las primeras dos capas, y una unica nerona para la ultima capa.

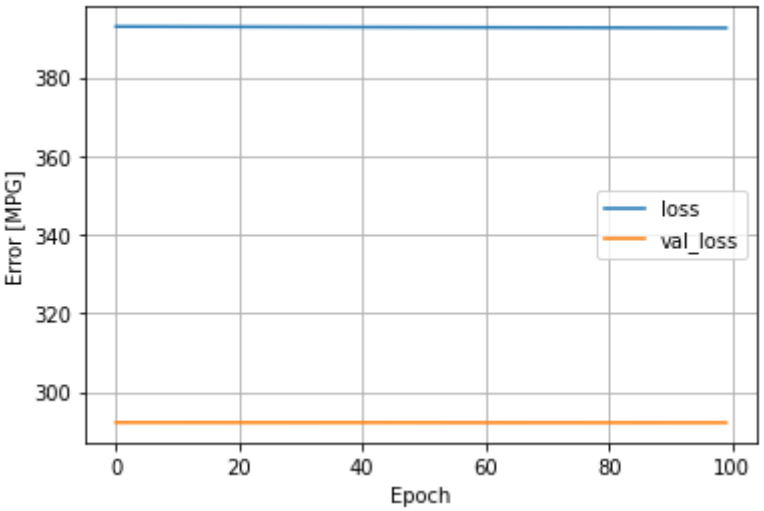
- Red neuronal grande

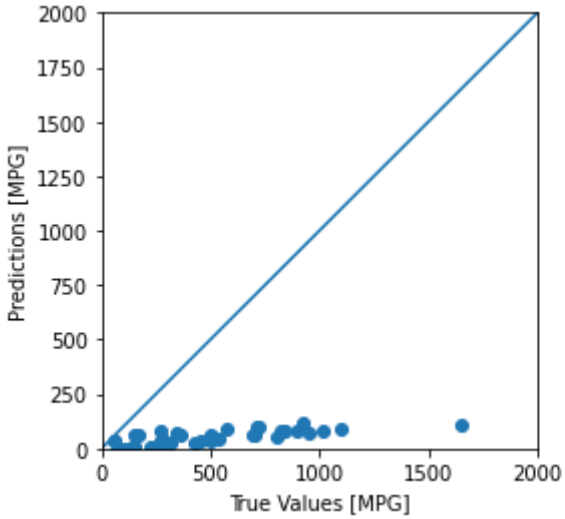
  Se implmento una red neuronal de seis capas, con 64 neronas y activacion mediante relu para las primeras tres capas, dos capas de 5 neuronas con activacion mediante relu, y una unica nerona para la ultima capa.

Metricas

- Regresion lineal:

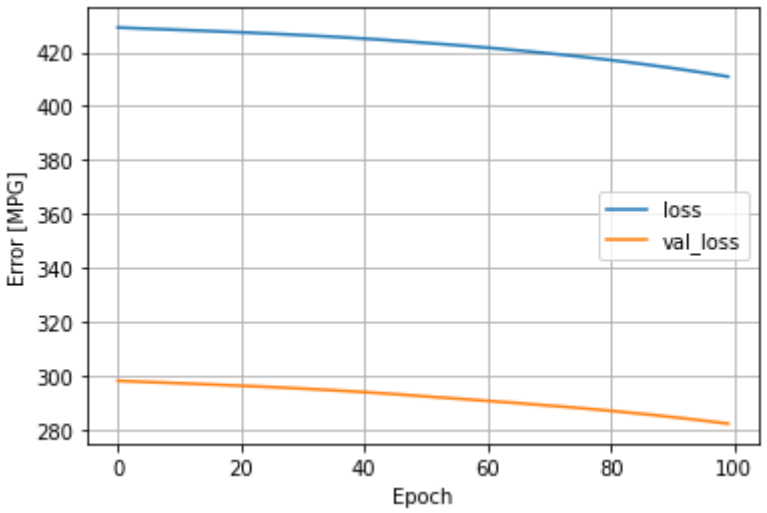
Loss: 358.95, Acurracy: 0.0.0

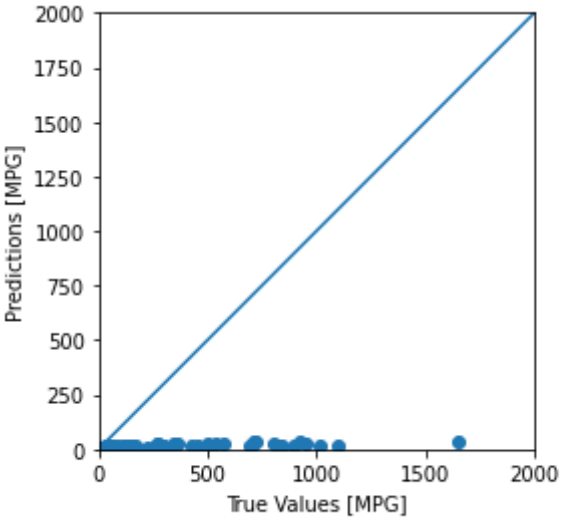




- Red neuronal diminuta:

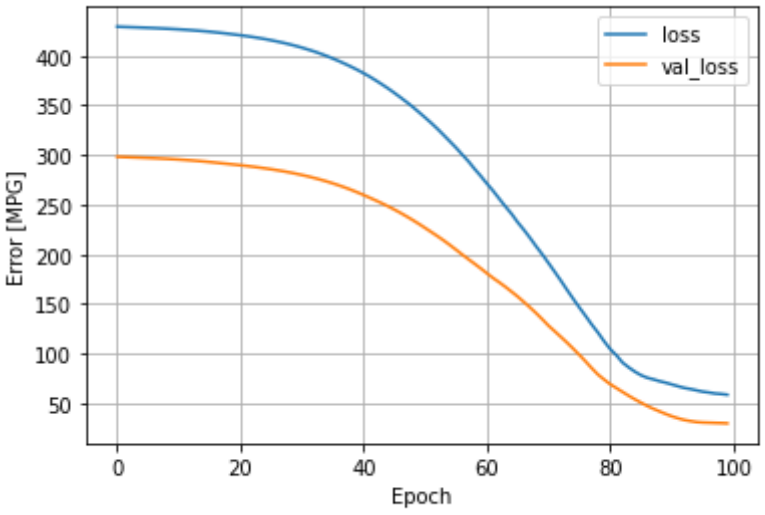
Loss: 358.373.53, Acurracy: 0.0.0

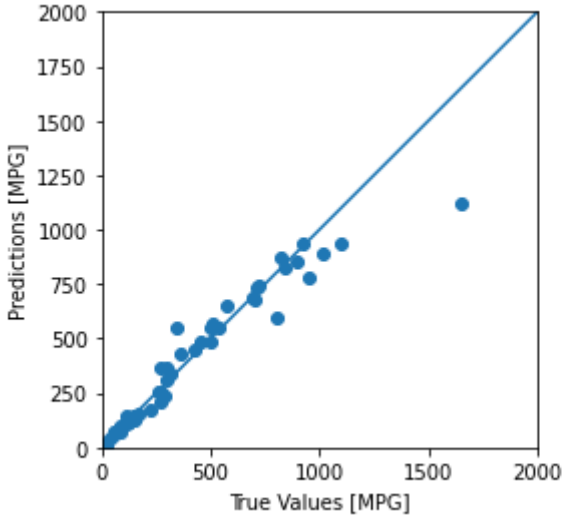




- Red neuronal pequeña:

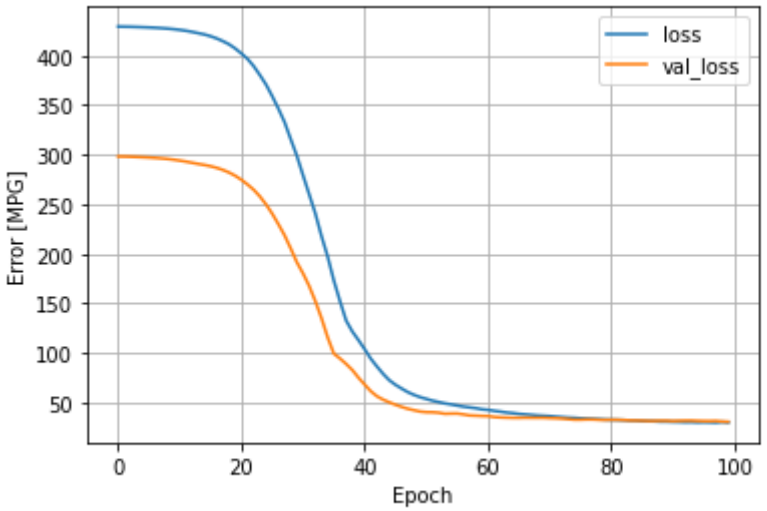
  Loss: 47.56, Acurracy: 0.0.0

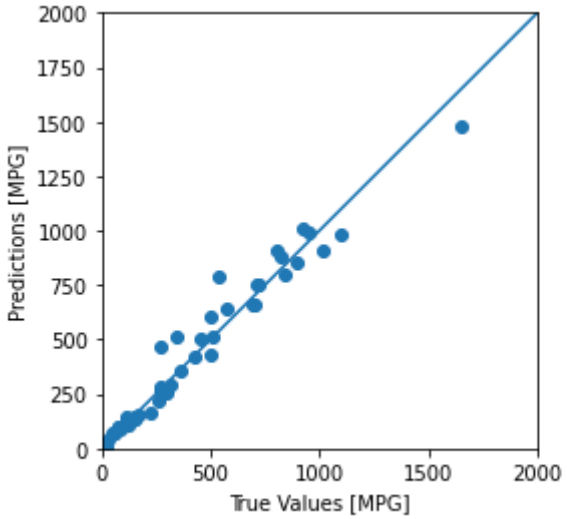




- Red neuronal grande:

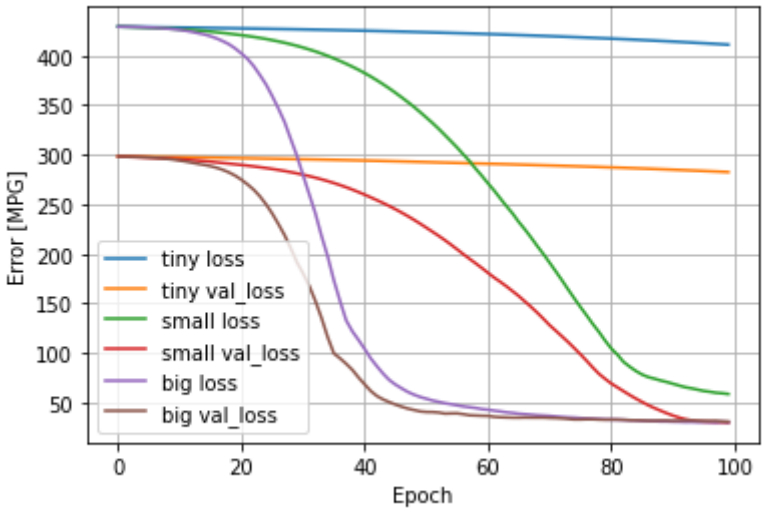
  Loss: 44.68, Acurracy: 0.0.0





Analisis

Tanto el modelo de regresion lineal como la red neuronal diminuta presentan underfiting, bias alto, y varianza baja. Mientrastando las redes neuronales pequeña y grande no presentan ni overfiting, ni underfiting, si bien ambas poseen un bias bajo, la red neuronal pequeña presenta una varianza un poco mas alta. Por lo tanto el modelo que mejor rendimiento tiene para predecir el peso de un pez en base a sus caracteristicas fue la red neuronal grande con un puntaje a base de r cuadrada de .96.

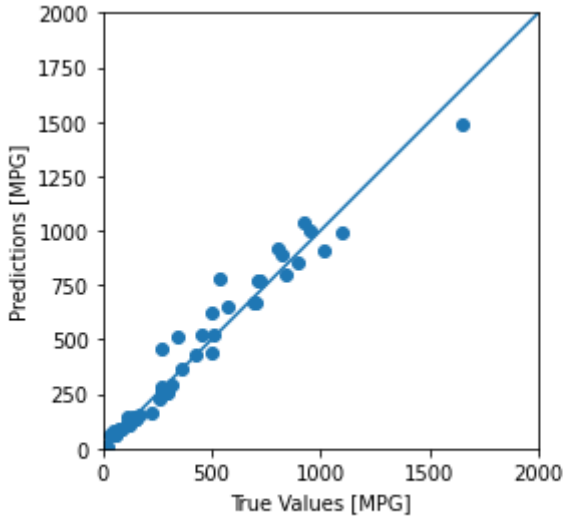


Regularizacion

Por el analisis antes mencionado se escogio la red neuronal grande para regularizar. Se nalizaron dos regularizacion es manera individual.

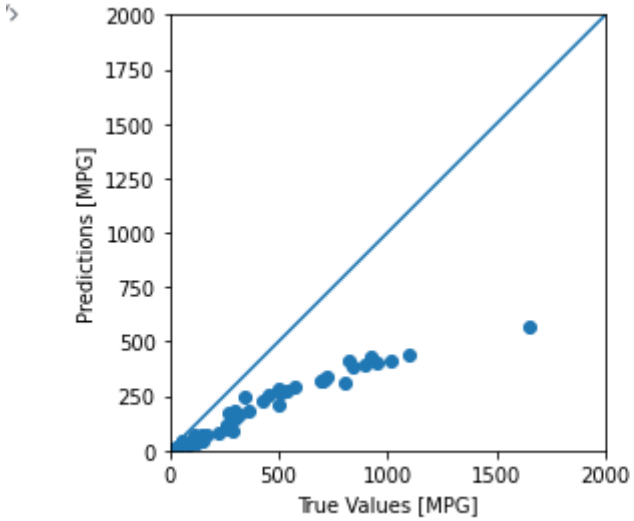
- Regularizacion mediante L2:

  Se implemento regularizacion de tipo L2 de 0.001 a las primeras cinco capas del modelo, con el cual se obtuvo un puntaje en base a r cuadrada de .96 y loss de 45.38.



- Regularizacion mediante dropout:

  Se implemento regularizacion de tipo dropout en las primeras cinco capas del modelo, con el cual se obtuvo un puntaje en base a r cuadrada de .26, y loss de 213.38, aumentando el bias y la varianza.



Resultados

Comparando las veriones del modelo de red neuronal grande, se puede conlcuir que tanto por el promedio del error absoluto, como por puntaje en r cuadrada la mejor version es la original sin regularizar

Conclusiones

Como apaprece en la guia de tensorflow al momento de ir generando modelos de redes neuronales lo mejor siempre sera iniciar con un modelo bastante sencillo e ir incrementando la complejida para ver como se comportan los cambios, para finalmente ver si es posible mejorarlo con regularizaciones, si bien en este caso no se pudo mejorar en otras situaciones podrian mejorar considerablementes, como incluso seguir probando con otras combinaciones de regularizacion podria mejorar el modelo.