Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Querétaro

TC3006C. Inteligencia artificial avanzada para la ciencia de datos I
Grupo 101

Momento de Retroalimentación Módulo 2 :

"Análisis y Reporte Sobre el Desempeño del Modelo"

Evidencia presentada por el estudiante :

Javier Suarez Duran A01707380

Profesores :

Benjamín Valdés Aguirre
José Antonio Cantoral Ceballos
Carlos Alberto Dorantes Dosamantes
Denisse L. Maldonado Flores
Alejandro Fernández Vilchis

Fecha de entrega :

Domingo 10 de Septiembre de 2023

Resumen — En este reporte se analizará el desempeño del modelo de machine learning desarrollado en la actividad "Momento de Retroalimentación: Módulo 2: Uso de framework o biblioteca de aprendizaje máquina para la implementación de una solución." para conocer del desempeño conseguido por el modelo en la actividad y mejorar en la medida de lo posible. El programa con el modelo utiliza el lenguaje de programación "Python" y puede ser encontrado en el siguiente repositorio de GitHub: https://github.com/Javier1257/T3006C-ML_Model_With_Specialized_Framework.git
Palabras clave — *modelo, análisis, desempeño, machine learning, GitHub.*

I. Introducción

El modelo a ser analizado es una red neuronal multicapa cuya función es determinar qué pasajeros del gran barco "Titanic" sobrevivieron al fatídico suceso por el que pasó el navío; por lo tanto utilizando métricas y gráficas se pondrá a prueba el modelo de manera que se pueda medir con la mayor certeza posible el desempeño del mismo con el objetivo de conocer si ¿hay mejora posible?, o si es así ¿cuáles deben de ser las implementaciones por ser agregadas al modelo para que mejore?.

II. Descripción del modelo

Para poder analizar el modelo lo principal es conocerlo, así que partiendo de esta base retomaremos lo mencionado anteriormente lo cual es que se trata de una red neuronal multicapa; el modelo en total cuenta con cuatro capas, una capa entrada, una de salida y dos capas internas; entrando en detalle del modelo comenzamos por la primera capa que es la de entrada la cual cuenta con nueve neuronas dado que para el problema que se está resolviendo es el número de variables que se determinaron como entrada después de realizar el respectivo ETL (Explore - Transform - Load) al set de datos del problema, también cabe mencionar que los datos fueron estandarizados y que la distribución del mismo fue de un 80% entrenamiento, 10% validación y 10% para evaluación pero continuando con la descripción de la capa también nos encontramos con un bias y que utiliza "relu" como función de activación; ahora para el caso de la segunda capa esta tiene 32 neuronas, "relu" una vez más como función de activación, y "Regularización L2" para evitar overfitting en el modelo; en la tercera capa se repite el mismo proceso que en su predecesora, 32 neuronas, "relu" como función de activación y "Regularización L2" para evitar overfitting en el modelo; por último la capa de salida solo contiene una única neura y utiliza una sigmoide como función de activación.

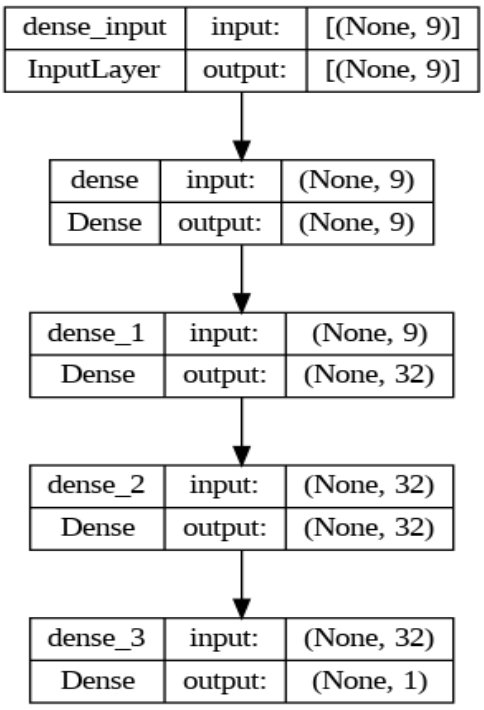
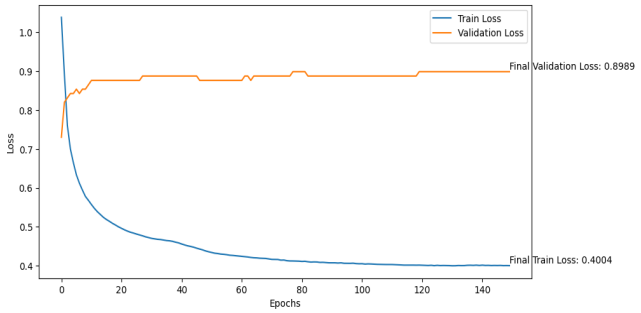


Imagen representativa del Modelo Generada por Código

El modelo fue entrenado por medio de "MiniBatches" de 32, a través de 150 épocas con un Learning Rate de 0.001, función de optimización "Adam", un valor de Regularización L2 de 0.01 y una función de pérdida "Binary - Cross Entropy" dado que solo se espera predecir una clase

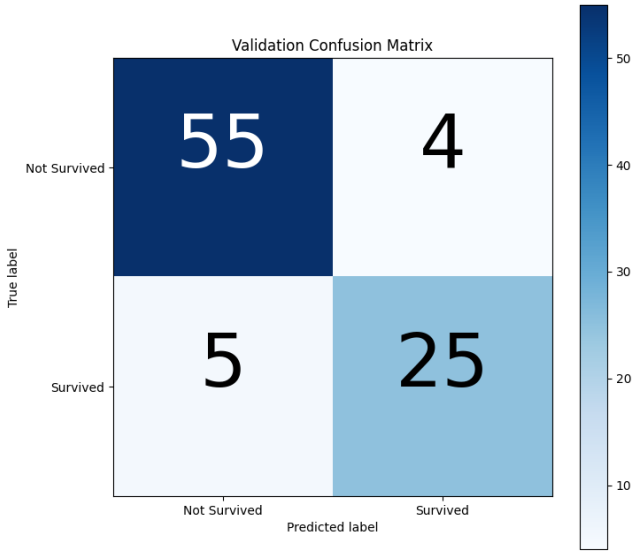
III. Análisis del Modelo

Tras el entrenamiento del modelo se generó la siguiente gráfica de pérdida en relación al set de datos de entrenamiento y validación:

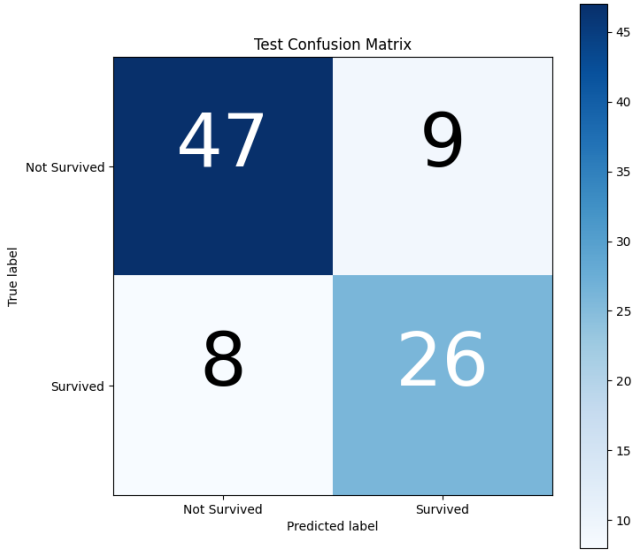


Gráfica de pérdidas generada por código

En el gráfico se puede observar que para este primer entrenamiento del modelo hay una alta varianza dado que la brecha que existe entre ambas líneas es grande, prácticamente de 0.5; con esta información se puede inferir que el modelo hizo overfitting dado el comportamiento que tiene la línea que representa la pérdida en la validación ("Validation Loss"), la cual aumenta en vez de bajar a diferencia de la pérdida en el entrenamiento ("Train Loss") la cual baja constantemente. Continuando con el análisis de desempeño se realizó una matriz de confusión para el set de validación y el de pruebas:



Matriz de Confusión generada por código



Matriz de Confusión generada por código

Con las cuales se consiguieron las siguientes métricas:

```
Validation Metrics
Precision:  0.86
Accuracy:  0.90
Recall:  0.83
F1:  0.85
```

```
Test Metrics
Precision:  0.74
Accuracy:  0.81
Recall:  0.76
F1:  0.75
```

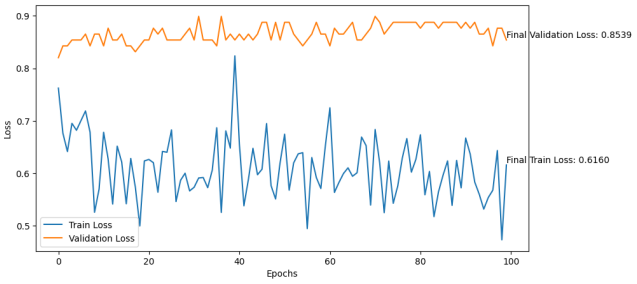
Valores calculados en el código

Por lo tanto como se puede observar en las métricas de desempeño del modelo es claro que no está aprendiendo adecuadamente dado el puntaje alcanzado en sus respectivas pruebas, por lo que se puede confirmar que efectivamente el modelo hizo "overfitting"; importante mencionar también que el problema de estudio contiene más valores falsos que verdaderos en las etiquetas de la clase, cosa que hace sentido en relación al contexto del problema ya que sobrevivió poca gente en comparación con la que estaba a bordo en el barco, así que conociendo esta información se puede inferir que el modelo también tiene un sesgo hacia generar más respuestas falsas que positivas, sin embargo no hay suficiente información en las pruebas realizadas para determinar con completa certeza el nivel de sesgo en el modelo por lo que sin eliminar la posibilidad de su existencia en el este se tomará y definirá como un sesgo de nivel medio.

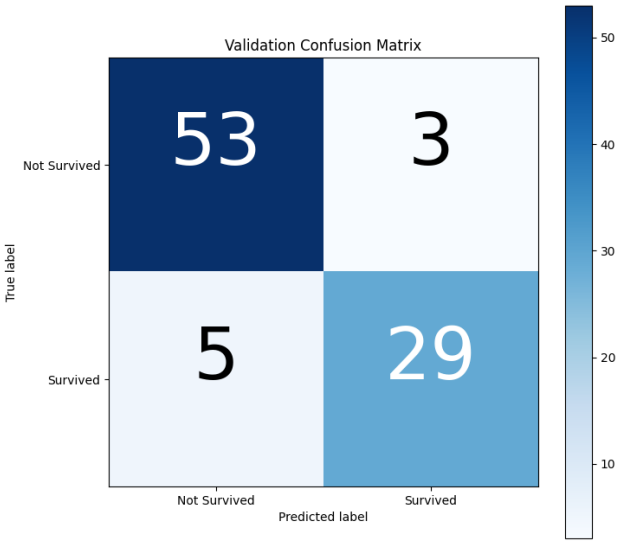
IV. Mejora

En respuesta al análisis realizado al modelo, el siguiente paso a realizar es tomar las medidas necesarias para mejorar su desempeño, por lo que tomando en cuenta que el modelo tiene una alta varianza, sesgo medio y que hizo overfitting lo principal es reducir su complejidad para evitar el problema de obtener la alta varianza y el overfitting, lo que deja por resolver el problema en el posible sesgo, el cual podría ser resuelto aumentando los datos de entrenamiento sin embargo el set de datos no es lo suficientemente grande para ello por lo que es una prueba que se continuara en otro momento, por lo pronto lo mejor que se puede se puede hacer para evitarlo y hacer que aprenda adecuadamente el modelo será modificar los hiperparametros hasta encontrar el punto adecuado en el que el modelo aprenda satisfactoriamente a reconocer los patrones dentro del set de datos.

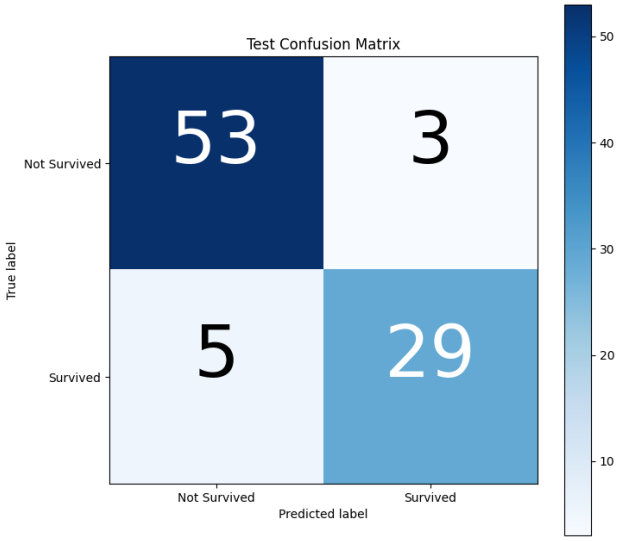
Por lo que después de los cambios pertinentes se consiguió lo siguiente:



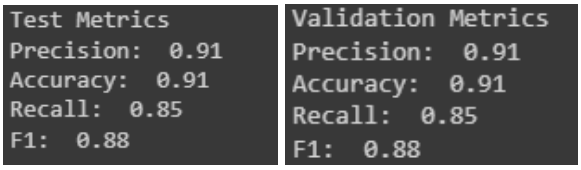
Gráfica de pérdida generada por código



Matriz de confusión generada por código



Matriz de confusión generada por código



Valores calculados en el código

Tras reducir la complejidad del modelo reduciendo el número de neuronas en las capas internas, agregar "dropout" y alterar los hiperparametros se consiguió que el modelo generalice mejor los datos, sin embargo a pesar de esto aún parece haber varianza en él según indica la gráfica de pérdida; pero a pesar de esto el modelo logra mejorar y conseguir mejores predicciones, el único problema sigue siendo la falta de datos para entrenar el modelo sin embargo de momento no han sido necesarios por lo que el desempeño conseguido al momento es suficiente para desempeñar un buen trabajo resolviendo el problema para el que fue entrenado.

V. Conclusión

En conclusión el proceso realizado durante el desarrollo del análisis ha sido importante para estudiar y comprender más acerca del modelo en este caso una red neuronal multicapa, sin embargo el proceso aplica para todos los modelos de machine learning de manera que se puede llegar a obtener los mejores resultados posibles y así poder desplegar estas importantes soluciones que facilitan la realización de todo tipo de tareas y problemas; por lo que conocer las métricas y gráficas que nos ayudan a visualizar de mejor manera lo que pasa dentro del modelo es de vital importancia para su continua mejora de manera que se pueden implementar los cambios necesarios al modelo para que este mejore.

VI. Anexos

Github del Modelo con la Red Neuronal Analizada:
https://github.com/Javier1257/T3006C-ML_Model_With_Specialized_Framework.git