Análisis y Reporte sobre el desempeño del modelo. Entrega final

Mariana Pérez Carmona A01731813

18/09/2022

Resumen

Para el análisis sobre el desempeño de este modelo he decidido utilizar la implementación del regresor lineal que se obtiene con ayuda de la librería scikit-learn, a partir del desempeño de esta implementación se generan diversas conclusiones al evaluar su desempeño y el proceso por el cual se han obtenido las predicciones requeridas.

1. Separación y evaluación del modelo con un conjunto de prueba y un conjunto de validación (Train/Test/Validation).

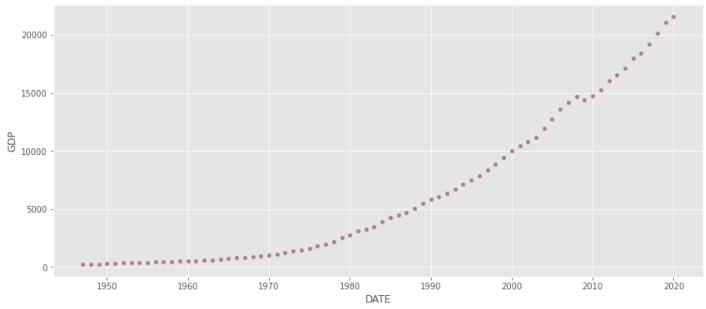

Figura 1: Gráfico PIB vs Año

En la gráfica de la Figura 1 podemos analizar en primer lugar un comportamiento exponencial o al menos similar a este del PIB de Estados Unidos a lo largo del tiempo. Las siguientes líneas del código nos muestran la separación de los datos en la variable independiente y y la variable dependiente X.

```
X=datos['DATE'].values.reshape(-1,1)
y=datos['GDP'].values
```

Para dividir el total de los datos (del año 1947 al año 2021) seleccioné los primeros 60 registros como datos de entrenamiento y lo últimos 14 como datos de prueba, esto porque me pareció mejor para el modelo que se entrenara con un orden y no que revolviera los valores de la variable años.

```
X_train=X[0:60]
X_test=X[60:]
y_train=y[0:60]
y_test=y[60:]
```

2. Diagnóstico y explicación el grado de bias o sesgo

El modelo que se obtiene de la implementación de los valores de entrenamiento es el siguiente:

$$GDP = 203{,}43Year - 398350{,}82$$

Obteniendo datos que nos dan la siguiente gráfica:

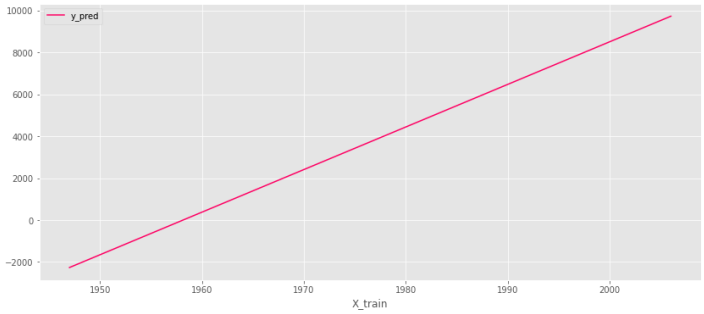

Figura 2: Modelo Estimado

El error de bias o de sesgo se refiere a la diferencia entre los valores verdaderos y los valores predichos por el modelo, en este sentido, es común que los regresores lineales no se desempeñen siempre de la mejor manera en este sentido puesto que no todos los datos siguen un crecimiento lineal.

En este sentido para identificar el grado de sesgo debemos comparar los datos que se obtienen del modelo generado y los datos reales, para ello se puede utilizar la métrica conocida como *Error Cuadrático Medio* y detectar en base a ella el grado de sesgo.

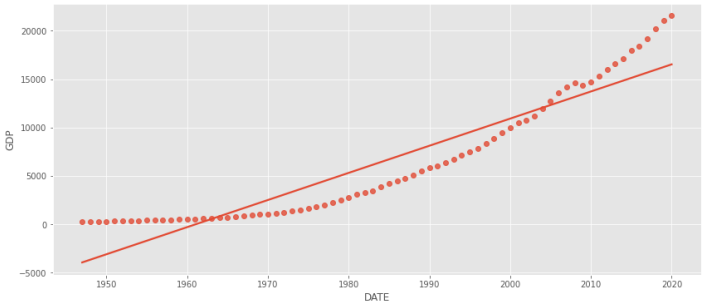

Figura 3: Valores Predichos vs Reales

```
print("Mean squared error: %.2f" % mean_squared_error(y_train, y_pred))
```

El resultado es $MSE = 2,198,834{,}08$, tomando en cuenta que las cantidades correspondientes a la variable GDP toman valores que van desde 243 a $21,539$ podemos entender que las cantidades obtenidas sean grandes, sin embargo al mirar gráficamente los resultados, nos percatamos de que la línea del modelo no se ajusta muy bien a los datos reales, por lo que el grado de sesgo puede llegar a ser alto para futuros valores.

3. Diagnóstico y explicación el grado de varianza.

Derivado de la separación entre los datos de entrenamiento y los datos de prueba, el algoritmo puede llegar a presentar variaciones entre el modelo que elige dependiendo de los datos con los que se entrena, un alto grado de varianza podría significar que el algoritmo se puede ver fuertemente influenciado por los datos de entrenamiento y por tanto no sea bueno generalizando.

Para detectar la varianza se puede utilizar la métrica de estimación de errores r^2 y esta se puede definir como la proporción de varianza de la variable dependiente que se puede predecir con la variable independiente.

```
print('Variance score: %.2f' % r2_score(y_train, y_pred))
```

EL valor obtenido es de 0.85, lo que significa que 85 % de la varianza de la variable GDP se puede estimar con la variable Date, esto es un alto grado de varianza por lo que ambas variables presentan correlación.

4. Diagnóstico y explicación el nivel de ajuste del modelo.

Para definir si un modelo esta sobre o sub ajustado debemos visualizar su comportamiento con los datos de entrenamiento y luego con nuevos datos, en este caso las métricas obtenidas con los datos de prueba son mucho peores que las que se obtienen con los datos de entrenamiento. Esto sugiere un sobre ajuste en el modelo y debe ser corregido para crear nuevas predicciones.

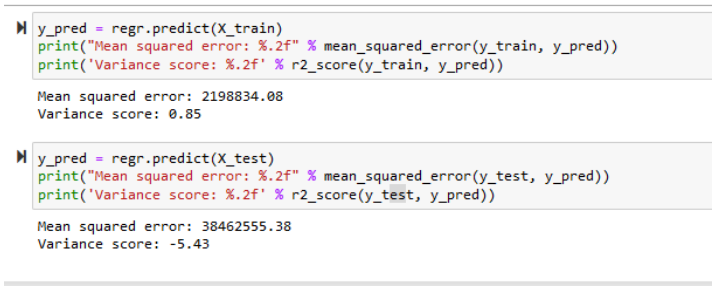

Figura 4: Evaluación datos entrenamiento y prueba

5. Mejoras en el modelo

La regresión lineal no tiene muchos parámetros que se puedan ajustar, debido a ello para mejorar el modelo decidí optar por utilizar la función $train_test_split$ para separar los datos de entrenamiento y prueba, con ello mejore el valor r^2 a 88 %.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10)
```