Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 03/04/2022
Practica No.8 Regresión Lineal Múltiple.

Objetivo.

Se desea obtener el pronóstico de la saturación de aceite remanente (ROS, Residual Oil Saturation).

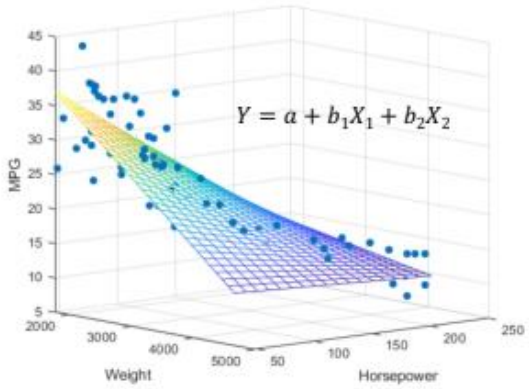
Características.

Se tienen mediciones de registros geofísicos convencionales: RC1 (Registro Neutrón), RC2 (Registro Sónico), RC3 (Registro Densidad-Neutrón) y RC4 (Registro Densidad -corregido por arcilla-).

La regresión lineal múltiple es un algoritmo que busca calcular una salida Y (Una variable dependiente) en función de múltiples entradas (Variables dependientes) calculando la ecuación que genere la recta o el hiperplano el cual minimiza la distancia entre cada uno de los puntos y el hiperplano ajustado.

$$\hat{Y} = a + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n + \mu$$

Donde μ, es el residuo del modelo que representa la diferencia entre el punto pronosticado y la observación real. Es importante considerar que al ser un algoritmo de aprendizaje supervisado los datos tienen que estar etiquetados, es decir, que todos deberán tener un valor real de la variable dependiente el cual se va a utilizar para entrenar al modelo.



Desarrollo.

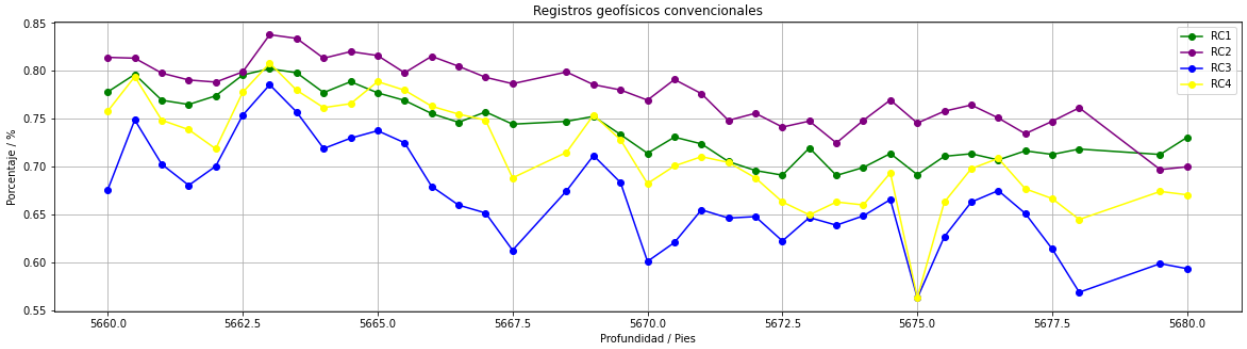
Como primer paso tenemos la importación de las librerías necesarias para trabajar con el conjunto de datos de los pacientes a segmentar. `pandas` para la manipulación de los datos, `numpy` para el manejo de matrices, `matplotlib` y `seaborn` para la visualización de estos datos mediante gráficos de diferente tipo dependiendo del análisis.

Después tenemos que hacer la lectura de nuestros datos los cuales están relacionados a la medición aproximada de la saturación de aceite remanente a cierta profundidad. Por ejemplo, para los 5661 m se van a tener 4 mediciones de la cantidad aproximada de aceite a esa profundidad. La idea es juntar varias mediciones las cuales nos servirán para hacer una regresión mas robusta que condicione esta variable.

Tenemos que leer los datos con ayuda de pandas el cual nos genera un `dataframe` con 38 registros y 5 columnas, la primera de ellas es la profundidad y las otras 4 corresponden a las diferentes formas de medir la cantidad de aceite a esa profundidad.

	Profundidad	RC1	RC2	RC3	RC4
0	5660.0	0.777924	0.814029	0.675698	0.757842
1	5660.5	0.796239	0.813167	0.748670	0.793872
2	5661.0	0.769231	0.797562	0.702285	0.748362
3	5661.5	0.764774	0.790365	0.680289	0.738451
4	5662.0	0.773813	0.788184	0.700248	0.718462
5	5662.5	0.795627	0.798850	0.753472	0.777537
6	5663.0	0.802155	0.837717	0.785441	0.807957
7	5663.5	0.797878	0.833851	0.756847	0.779641
8	5664.0	0.777206	0.813117	0.718713	0.761454
9	5664.5	0.788604	0.820041	0.729582	0.765600

A continuación, se muestra la grafica en la cual podemos ver la profundidad del pozo en el eje X y el porcentaje de aceite residual a esa profundidad en el eje Y.



Una vez que se tiene bien identificada la forma de los datos podemos aplicar el modelo, para ello nos ayudaremos de las librerías de `linear_model` para generar el modelo, `mean_squared_error`, `max_error` y `r2_score` los cuales nos van a ayudar a generar las métricas para evaluar la efectividad del modelo una vez lo tengamos.

Para ello seleccionaremos la variable dependiente del modelo como la cantidad en porcentaje de aceite residual y las variables independientes del modelo las cuales serán `profundidad, RC1, RC2 y RC3` que formaran parte del conjunto de entrenamiento y luego esta `RC4` la cual servirá como etiqueta del valor real debido a que es la medición que mas aproximada esta a una cantidad real de aceite a cierta profundidad por la forma en la que se mide eliminando las impurezas.

En este modelo no es necesario estandarizar los datos debido a que la mayoría se presentan ya con el mismo rango evitando así que se tengan sesgos en modelo generado para el pronóstico.

```
X_train = np.array(RGeofisicos[['Profundidad', 'RC1', 'RC2','RC3']])
pd.DataFrame(X_train)
Y_train = np.array(RGeofisicos[['RC4']])
pd.DataFrame(Y_train)
```

Se dividen los conjuntos de entrenamiento como arreglos de `numpy` para proceder a entrenar el modelo.

```
RLMultiple = linear_model.LinearRegression()
RLMultiple.fit(X_train, Y_train)
```

Con nuestro modelo generado como un objeto de la clase `linear_model.LinearRegression` (Al cual llamamos RLMultiple) podemos utilizar el método `fit` pasando como datos los parámetros de los arreglos con los conjuntos de entrenamiento. Después de esto podemos utilizar el modelo de regresión lineal múltiple generado a partir de estos datos con el método `predict`.

```
Y_pronostico = RLMultiple.predict(X_train)
pd.DataFrame(Y_pronostico)
```

Esto nos generara los valores que se pronostican para cada uno de los registros con los que se entreno el modelo, sin embargo, debemos recordar que estos valores presentan cierto grado de error al no ajustarse totalmente al hiperplano. Esto lo podemos ver mas claramente si mostramos el dataframe generado y agregamos una nueva columna a la cual llamamos pronostico.

	Profundidad	RC1	RC2	RC3	RC4	Pronostico
0	5660.0	0.777924	0.814029	0.675698	0.757842	0.747294
1	5660.5	0.796239	0.813167	0.748670	0.793872	0.792029
2	5661.0	0.769231	0.797562	0.702285	0.748362	0.752073
3	5661.5	0.764774	0.790365	0.680289	0.738451	0.737382
4	5662.0	0.773813	0.788184	0.700248	0.718462	0.751189
5	5662.5	0.795627	0.798850	0.753472	0.777537	0.790661
6	5663.0	0.802155	0.837717	0.785441	0.807957	0.818408

Con los atributos `coef_` e `intercept_` podemos consultar cuales son los coeficientes de nuestro hiperplano el cual se va a ajustar a la mayoría de los registros siguiendo la siguiente ecuación.

$$\hat{Y} = 0.2624 - 0.000075(Profundidad) + 0.5066(RC1) + 0.2275(RC2) + 0.4891(RC1)$$

Sin embargo, aun faltaría agregar el residuo del modelo el cual lo podemos obtener con ayuda de la clase `max_error` que se encuentra en el paquete `sklearn.metrics` de Sklearn.

A continuación, se muestra el modelo de regresión lineal múltiple utilizado para hacer las predicciones.

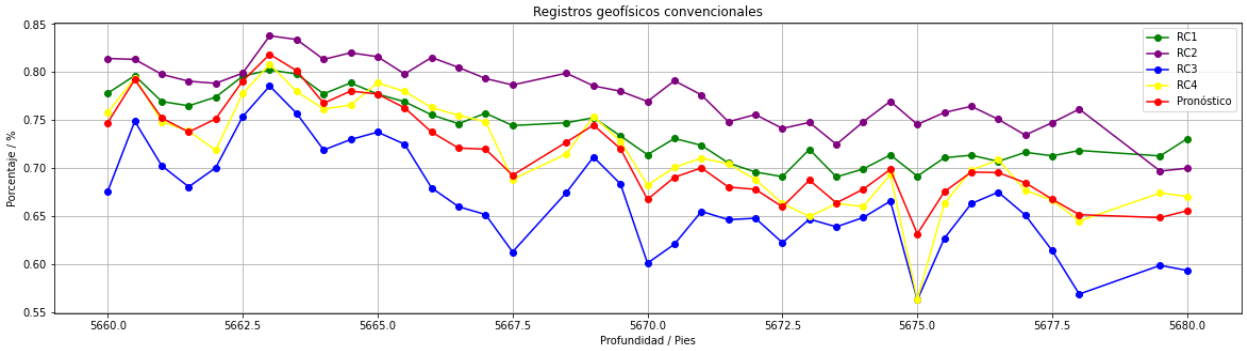
$$\hat{Y} = 0.2624 - 0.000075(Profundidad) + 0.5066(RC1) + 0.2275(RC2) + 0.4891(RC1) + 0.0684$$

Donde 0.0684 representa el residuo, es decir una medida general la cual nos dice que tan diferentes son los valores pronosticados de los valores reales.

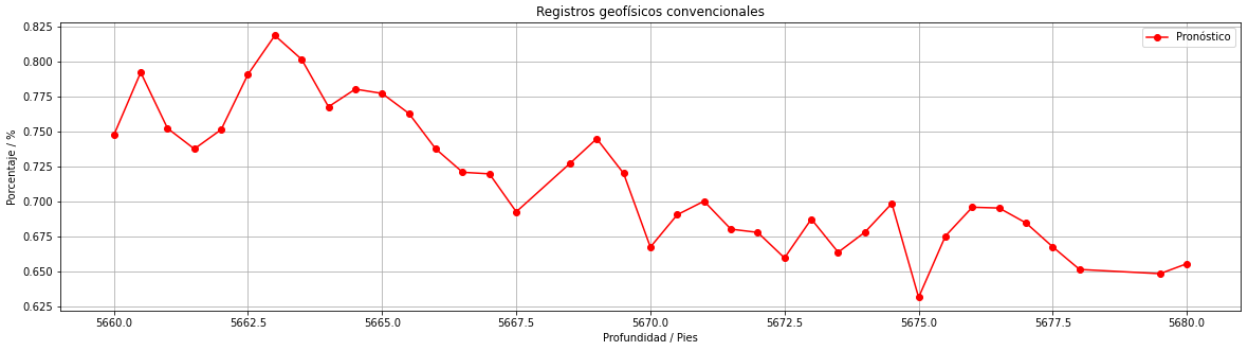
Además, podemos calcular el MSE y su raíz, el MRSE las cuales son dos métricas que nos indican en promedio que tantas unidades se alejan los valores pronosticados de los valores reales los cuales son 0.0004 y 0.0195 respectivamente. Estos valores se calcularon con la clase `mean_squared_error` de Sklearn.

Como ultima métrica se calculo el score que indica el porcentaje de precisión del modelo siendo un indicativo de su efectividad con nuevos valores el cual se logró de 85.81% que para el contexto en el que se está trabajando resulta ser un modelo muy efectivo.

A continuación, se muestra una grafica la cual sobrepone todas las formas de hacer la medición del aceite residual.



Podemos ver que el comportamiento de las predicciones hace sentido con las demás mediciones comportándose como un punto medio en la mayoría de las ocasiones. A continuación, se muestra solamente la grafica del pronostico la cual en su eje X indica la profundidad del pozo y en su eje Y la cantidad de aceite residual a esa profundidad.



Por último, es importante tener en cuenta que con este modelo podemos llegar a hacer extrapolaciones ingresando nuevos valores para cada una de las variables independientes que seleccionamos para el modelo en un inicio. Sin embargo, estas predicciones normalmente tienen un límite físico por ejemplo en este caso la profundidad del pozo la cual no puede ir mucho más allá de 5680 m. Para probar nuestro modelo ingresamos datos para una profundidad de 5680.5 m obteniendo un porcentaje de aceite residual de 62.7% el cual no se aleja mucho del valor real para los 5680 m (Apenas 0.5 metros menos) que es de 67%.

Conclusiones.

En esta práctica se logró generar un modelo de regresión lineal múltiple el cual ajusta un hiperplano a una serie de registros. Para esto fue necesario, primero, entender la estructura de los datos geofísicos con los cuales se estuvo trabajando para después hacer la selección de la variable dependiente (% de aceite residual) la cual va a estar en función de 2 o más variables independientes (Profundidad, RC1, RC2 y RC3). Debemos tener en cuenta que como estamos trabajando con un modelo de aprendizaje supervisado es necesario etiquetar con un valor real a cada uno de los datos para que precisamente se genere aprendizaje en el modelo. Una vez que generemos el modelo podremos ser capaces de hacer nuevos pronósticos con datos que el modelo nunca ha procesado, sin embargo, estos pronósticos presentan un cierto grado de incertidumbre el cual se busca reducir lo mas posible cuidando las métricas de rendimiento.

Es importante generar las métricas del modelo para conocer que tan efectivo puede llegar a ser, el residuo (0.0684) que es directamente la diferencia entre el valor pronosticado y el valor real generado un solo índice considerando cada registro. El error cuadrático medio y su raíz, MSE (0.004) y RMSE (0.0195) respectivamente que indican las unidades en promedio que las predicciones se alejan del valor real y por ultimo el Score del modelo que nos va a representar con un porcentaje que tan preciso resulto ser el modelo obteniendo un 85.81% que bajo este contexto en una métrica más que aceptable.

Fuentes.

pandas documentation — pandas 1.4.1 documentation. (2022). Pandas. Recuperado 2022, de https://pandas.pydata.org/docs/index.html#

User guide: contents. (2022). Scikit-Learn. Recuperado 2022, de https://scikit-learn.org/stable/user_guide.html