```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
df=pd.read_csv("MiningProcess_Flotation_Plant_Database.csv")
df
```

	date	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	...	Flotation Column 07 Air Flow	Flotation Column 01 Level	F C
0	2017-03-10 01:00:00	55,2	16,98	3019,53	557,434	395,713	10,0664	1,74	249,214	253,235	...	250,884	457,396	
1	2017-03-10 01:00:00	55,2	16,98	3024,41	563,965	397,383	10,0672	1,74	249,719	250,532	...	248,994	451,891	
2	2017-03-10 01:00:00	55,2	16,98	3043,46	568,054	399,668	10,068	1,74	249,741	247,874	...	248,071	451,24	
3	2017-03-10 01:00:00	55,2	16,98	3047,36	568,665	397,939	10,0689	1,74	249,917	254,487	...	251,147	452,441	
4	2017-03-10 01:00:00	55,2	16,98	3033,69	558,167	400,254	10,0697	1,74	250,203	252,136	...	248,928	452,441	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
737448	2017-09-09 23:00:00	49,75	23,2	2710,94	441,052	386,57	9,62129	1,65365	302,344	298,786	...	313,695	392,16	
737449	2017-09-09 23:00:00	49,75	23,2	2692,01	473,436	384,939	9,62063	1,65352	303,013	301,879	...	236,7	401,505	
737450	2017-09-09 23:00:00	49,75	23,2	2692,2	500,488	383,496	9,61874	1,65338	303,662	307,397	...	225,879	408,899	
737451	2017-09-09 23:00:00	49,75	23,2	1164,12	491,548	384,976	9,61686	1,65324	302,55	301,959	...	308,115	405,107	
737452	2017-09-09 23:00:00	49,75	23,2	1164,12	468,019	384,801	9,61497	1,6531	300,355	292,865	...	308,115	413,754	

737453 rows × 24 columns

```
df=df.drop(columns=['date'])
```

```
df.head()
```

	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	Flotation Column 03 Air Flow	...	Flotation Column 07 Air Flow	Flotation Column 01 Level	Flot Colu
0	55,2	16,98	3019,53	557,434	395,713	10,0664	1,74	249,214	253,235	250,576	...	250,884	457,396	43
1	55,2	16,98	3024,41	563,965	397,383	10,0672	1,74	249,719	250,532	250,862	...	248,994	451,891	4
2	55,2	16,98	3043,46	568,054	399,668	10,068	1,74	249,741	247,874	250,313	...	248,071	451,24	46

```
def convert(x):
 return float(x.replace(',','.'))
```

5 rows × 23 columns

```
for i in df.columns:
 df[i]=df[i].apply(lambda x: convert(x))
```

```
df.head()
```

	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	Flotation Column 03 Air Flow	...	Flotation Column 07 Air Flow	Flotation Column 01 Level	Flot Colu
0	55.2	16.98	3019.53	557.434	395.713	10.0664	1.74	249.214	253.235	250.576	...	250.884	457.396	43
1	55.2	16.98	3024.41	563.965	397.383	10.0672	1.74	249.719	250.532	250.862	...	248.994	451.891	42
2	55.2	16.98	3043.46	568.054	399.668	10.0680	1.74	249.741	247.874	250.313	...	248.071	451.240	46
3	55.2	16.98	3047.36	568.665	397.939	10.0689	1.74	249.917	254.487	250.049	...	251.147	452.441	45
4	55.2	16.98	3033.69	558.167	400.254	10.0697	1.74	250.203	252.136	249.895	...	248.928	452.441	45

5 rows × 23 columns

```
df.info()
```

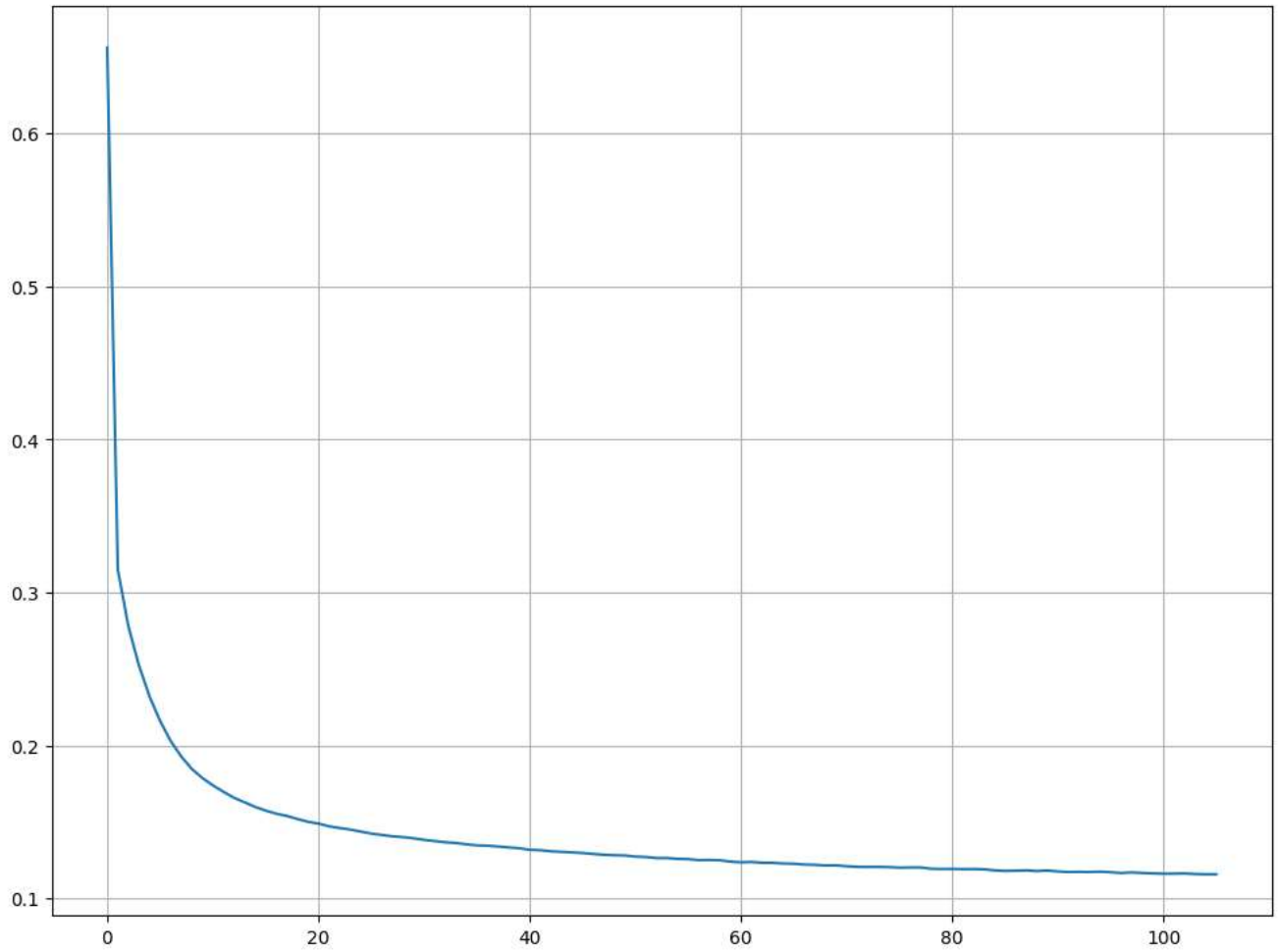
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 737453 entries, 0 to 737452
Data columns (total 23 columns):
 #   Column                        Non-Null Count   Dtype  
---  ------                        --------------   -----  
 0   % Iron Feed                   737453 non-null  float64
 1   % Silica Feed                 737453 non-null  float64
 2   Starch Flow                   737453 non-null  float64
 3   Amina Flow                    737453 non-null  float64
 4   Ore Pulp Flow                 737453 non-null  float64
 5   Ore Pulp pH                   737453 non-null  float64
 6   Ore Pulp Density              737453 non-null  float64
 7   Flotation Column 01 Air Flow  737453 non-null  float64
 8   Flotation Column 02 Air Flow  737453 non-null  float64
 9   Flotation Column 03 Air Flow  737453 non-null  float64
 10  Flotation Column 04 Air Flow  737453 non-null  float64
 11  Flotation Column 05 Air Flow  737453 non-null  float64
 12  Flotation Column 06 Air Flow  737453 non-null  float64
 13  Flotation Column 07 Air Flow  737453 non-null  float64
 14  Flotation Column 01 Level     737453 non-null  float64
 15  Flotation Column 02 Level     737453 non-null  float64
 16  Flotation Column 03 Level     737453 non-null  float64
 17  Flotation Column 04 Level     737453 non-null  float64
 18  Flotation Column 05 Level     737453 non-null  float64
 19  Flotation Column 06 Level     737453 non-null  float64
 20  Flotation Column 07 Level     737453 non-null  float64
 21  % Iron Concentrate            737453 non-null  float64
 22  % Silica Concentrate          737453 non-null  float64
dtypes: float64(23)
memory usage: 129.4 MB
```

```
df.head()
```

	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	Flotation Column 03 Air Flow	...	Flotation Column 07 Air Flow	Flotation Column 01 Level	Flot Colu
0	55.2	16.98	3019.53	557.434	395.713	10.0664	1.74	249.214	253.235	250.576	...	250.884	457.396	43
1	55.2	16.98	3024.41	563.965	397.383	10.0672	1.74	249.719	250.532	250.862	...	248.994	451.891	42
2	55.2	16.98	3043.46	568.054	399.668	10.0680	1.74	249.741	247.874	250.313	...	248.071	451.240	46
3	55.2	16.98	3047.36	568.665	397.939	10.0689	1.74	249.917	254.487	250.049	...	251.147	452.441	45

```
x=df.iloc[:,:-1]
y=df['% Silica Concentrate']


from sklearn.preprocessing import StandardScaler


sc=StandardScaler()
x=sc.fit_transform(x)


from sklearn.model_selection import train_test_split


x_train,x_test,y_train,y_test =train_test_split(x,y,test_size=0.3,random_state=1)


import tensorflow as tf
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Dropout
from sklearn.metrics import r2_score
es=EarlyStopping(monitor='val_loss',mode='min',verbose=1,patience=75,min_delta=0.2)
ann=Sequential()
ann.add(Dense(units=280,activation='relu'))
ann.add(Dense(units=1))
ann.compile(optimizer='adam',loss='mse',metrics=['accuracy'])
ann.fit(x_train,y_train,epochs=256,verbose=1,validation_data=(x_test,y_test),batch_size=1000,callbacks=[es])
loss=ann.history.history
loss_df=pd.DataFrame(loss)
plt.figure(figsize=(15,10))
plt.plot(loss_df['loss'])
plt.grid()
plt.show()
y_pred_test=ann.predict(x_test)
print("\nAccuracy:\n")
print(r2_score(y_test,y_pred_test))
```

```
517/517 [==============================] - 4s 8ms/step - loss: 0.1162 - accuracy: 0.0036 - val_loss: 0.1236 - val_accuracy: 0.0037
Epoch 103/256
517/517 [==============================] - 5s 10ms/step - loss: 0.1163 - accuracy: 0.0036 - val_loss: 0.1270 - val_accuracy: 0.0037
Epoch 104/256
517/517 [==============================] - 3s 6ms/step - loss: 0.1160 - accuracy: 0.0036 - val_loss: 0.1200 - val_accuracy: 0.0037
Epoch 105/256
517/517 [==============================] - 3s 6ms/step - loss: 0.1158 - accuracy: 0.0036 - val_loss: 0.1198 - val_accuracy: 0.0037
Epoch 106/256
517/517 [==============================] - 5s 10ms/step - loss: 0.1158 - accuracy: 0.0036 - val_loss: 0.1202 - val_accuracy: 0.0037
Epoch 106: early stopping
```




```
6914/6914 [==============================] - 15s 2ms/step

Accuracy:

0.9053582588311376
```