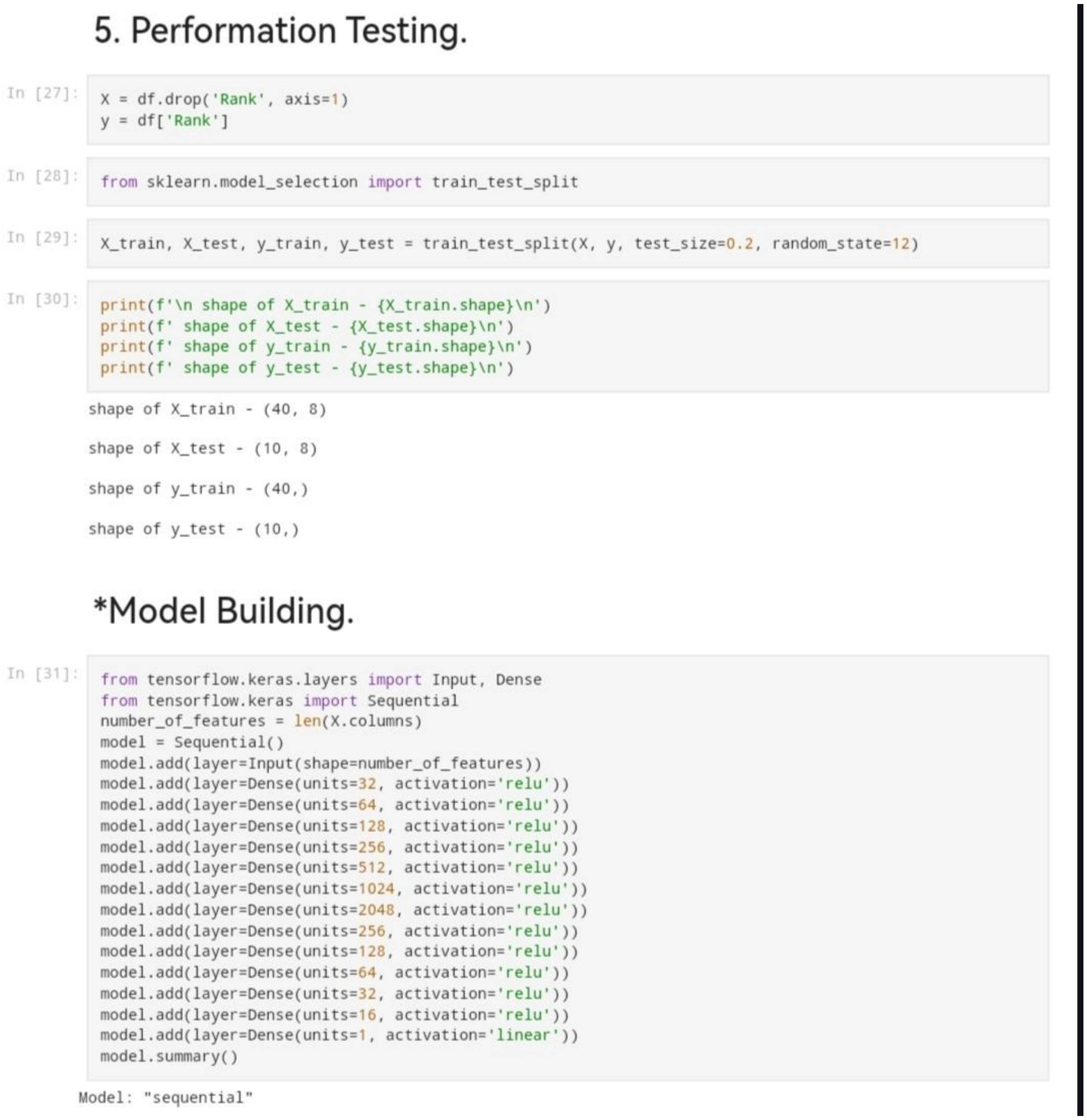
MODEL PERFORMANCE

Documentation:

DATE	03-11-2023
TEAM ID	NM2023TMID00283
PROJECT NAME	Subscriber Galore: Exploring the world's top youtube channels.

Model Performance:



5. Performation Testing.

```
In [27]: X = df.drop('Rank', axis=1)
         y = df['Rank']
```

```
In [28]: from sklearn.model_selection import train_test_split
```

```
In [29]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=12)
```

```
In [30]: print(f'\n shape of X_train - {X_train.shape}\n')
         print(f' shape of X_test - {X_test.shape}\n')
         print(f' shape of y_train - {y_train.shape}\n')
         print(f' shape of y_test - {y_test.shape}\n')
```

```
shape of X_train - (40, 8)

shape of X_test - (10, 8)

shape of y_train - (40,)

shape of y_test - (10,)
```

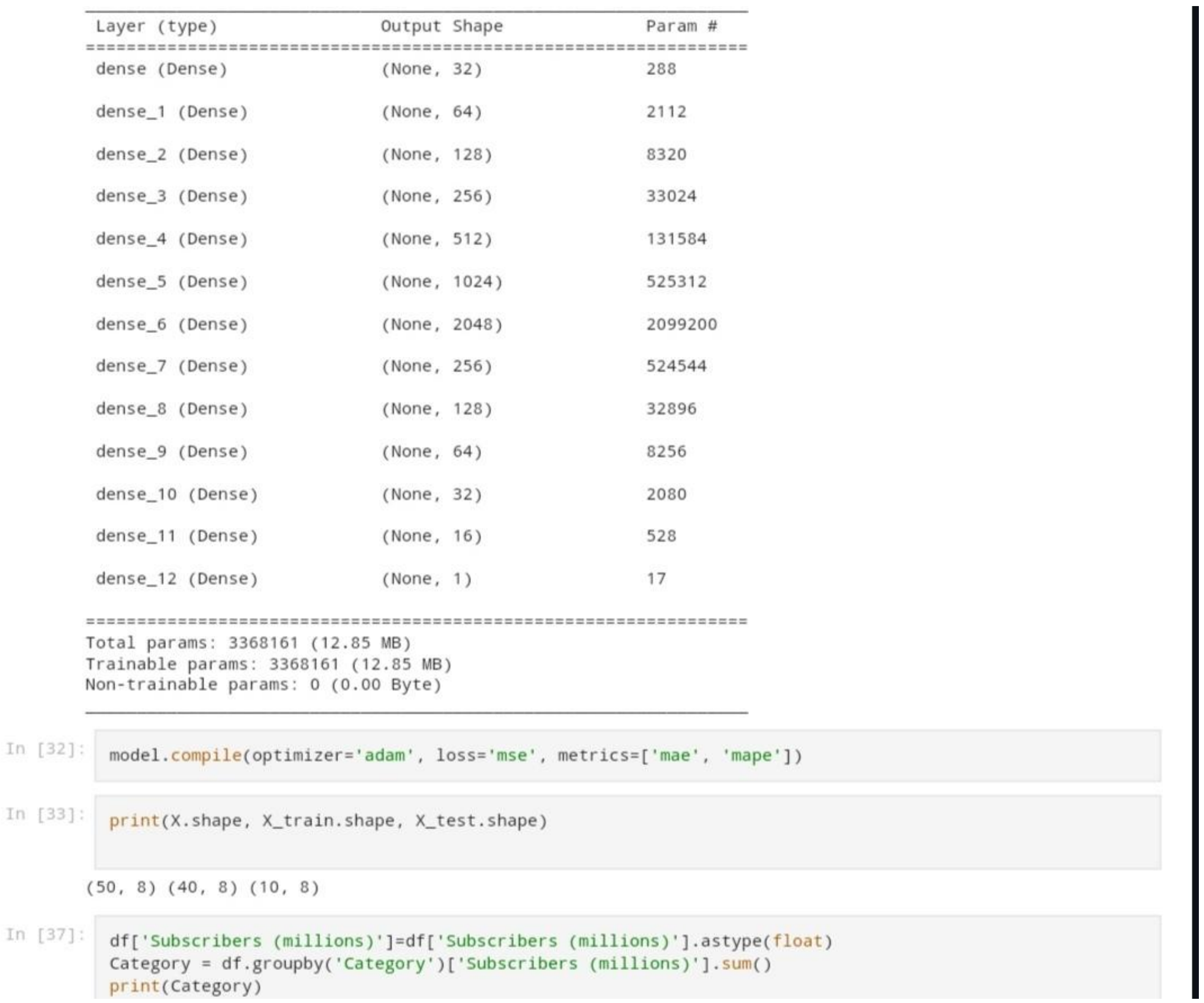
*Model Building.

```
In [31]: from tensorflow.keras.layers import Input, Dense
         from tensorflow.keras import Sequential
         number_of_features = len(X.columns)
         model = Sequential()
         model.add(layer=Input(shape=number_of_features))
         model.add(layer=Dense(units=32, activation='relu'))
         model.add(layer=Dense(units=64, activation='relu'))
         model.add(layer=Dense(units=128, activation='relu'))
         model.add(layer=Dense(units=256, activation='relu'))
         model.add(layer=Dense(units=512, activation='relu'))
         model.add(layer=Dense(units=1024, activation='relu'))
         model.add(layer=Dense(units=2048, activation='relu'))
         model.add(layer=Dense(units=256, activation='relu'))
         model.add(layer=Dense(units=128, activation='relu'))
         model.add(layer=Dense(units=64, activation='relu'))
         model.add(layer=Dense(units=32, activation='relu'))
         model.add(layer=Dense(units=16, activation='relu'))
         model.add(layer=Dense(units=1, activation='linear'))
         model.summary()
```

```
Model: "sequential"
```

MODEL PERFORMANCE

Documentation:



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 32)                288

 dense_1 (Dense)             (None, 64)                2112

 dense_2 (Dense)             (None, 128)               8320

 dense_3 (Dense)             (None, 256)               33024

 dense_4 (Dense)             (None, 512)               131584

 dense_5 (Dense)             (None, 1024)              525312

 dense_6 (Dense)             (None, 2048)              2099200

 dense_7 (Dense)             (None, 256)               524544

 dense_8 (Dense)             (None, 128)               32896

 dense_9 (Dense)             (None, 64)                8256

 dense_10 (Dense)            (None, 32)                2080

 dense_11 (Dense)            (None, 16)                528

 dense_12 (Dense)            (None, 1)                 17

=================================================================
Total params: 3368161 (12.85 MB)
Trainable params: 3368161 (12.85 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [32]:

```
model.compile(optimizer='adam', loss='mse', metrics=['mae', 'mape'])
```

In [33]:

```
print(X.shape, X_train.shape, X_test.shape)
```

```
(50, 8) (40, 8) (10, 8)
```

In [37]:

```
df['Subscribers (millions)']=df['Subscribers (millions)'].astype(float)
Category = df.groupby('Category')['Subscribers (millions)'].sum()
print(Category)
```