```
!pip install pydot -- in notebook
#sudo apt install graphviz -- in terminal
```

```
Requirement already satisfied: pydot in /usr/local/lib/python3.11/dist-packages (3.0.4)
ERROR: Could not find a version that satisfies the requirement in (from versions: none)
ERROR: No matching distribution found for in
```

```
import tensorflow as tf
tf.config.set_visible_devices([], 'GPU') #to use CPU instead of GPU
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from tqdm.notebook import tqdm
```

```
boston = tf.keras.datasets.boston_housing
```

```
dir(boston)
```

```
['__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__path__',
 '__spec__',
 'load_data']
```

```
boston_data = boston.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/boston_housing.npz
57026/57026 ━━━━━━━━━━━━━━━━━━━━ 0s 1us/step
```

```
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.boston_housing.load_data(path='boston_housing.npz', test_split=0.2, seed=42)
```

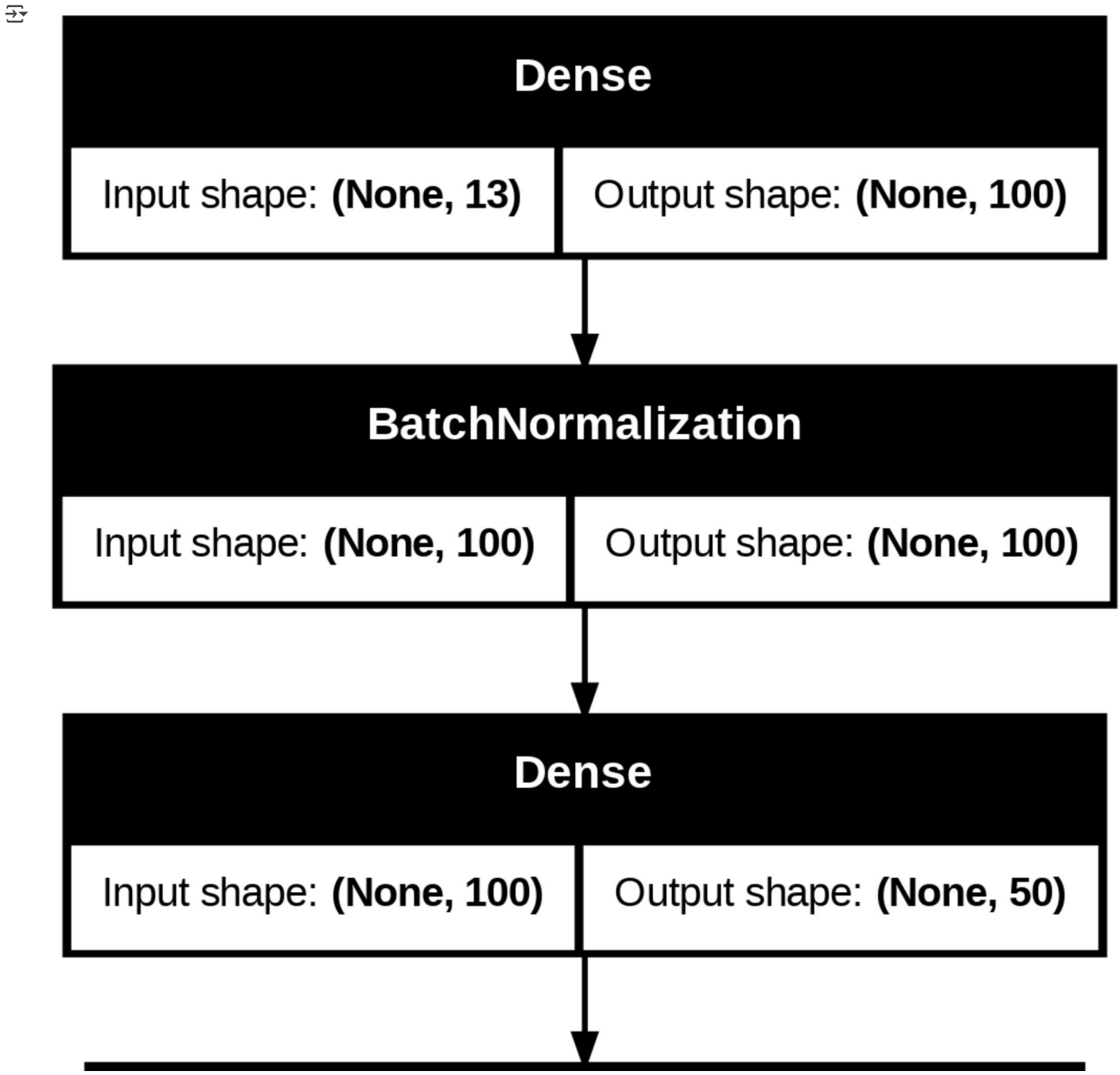
```
x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

```
((404, 13), (404,), (102, 13), (102,))
```

```
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
y_train_scaled = scaler.fit_transform(y_train.reshape(-1, 1))
y_test_scaled = scaler.transform(y_test.reshape(-1, 1))
```

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Input(shape=(13, ), name='input-layer'),
    tf.keras.layers.Dense(100, name='hidden-layer-2'),
    tf.keras.layers.BatchNormalization(name='hidden-layer-3'),
    tf.keras.layers.Dense(50, name='hidden-layer-4'),
    tf.keras.layers.Dense(1, name='output-layer')
])
```

```
tf.keras.utils.plot_model(model, show_shapes=True)
```




```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
hidden-layer-2 (Dense)	(None, 100)	1,400
hidden-layer-3 (BatchNormalization)	(None, 100)	400
hidden-layer-4 (Dense)	(None, 50)	5,050
output-layer (Dense)	(None, 1)	51

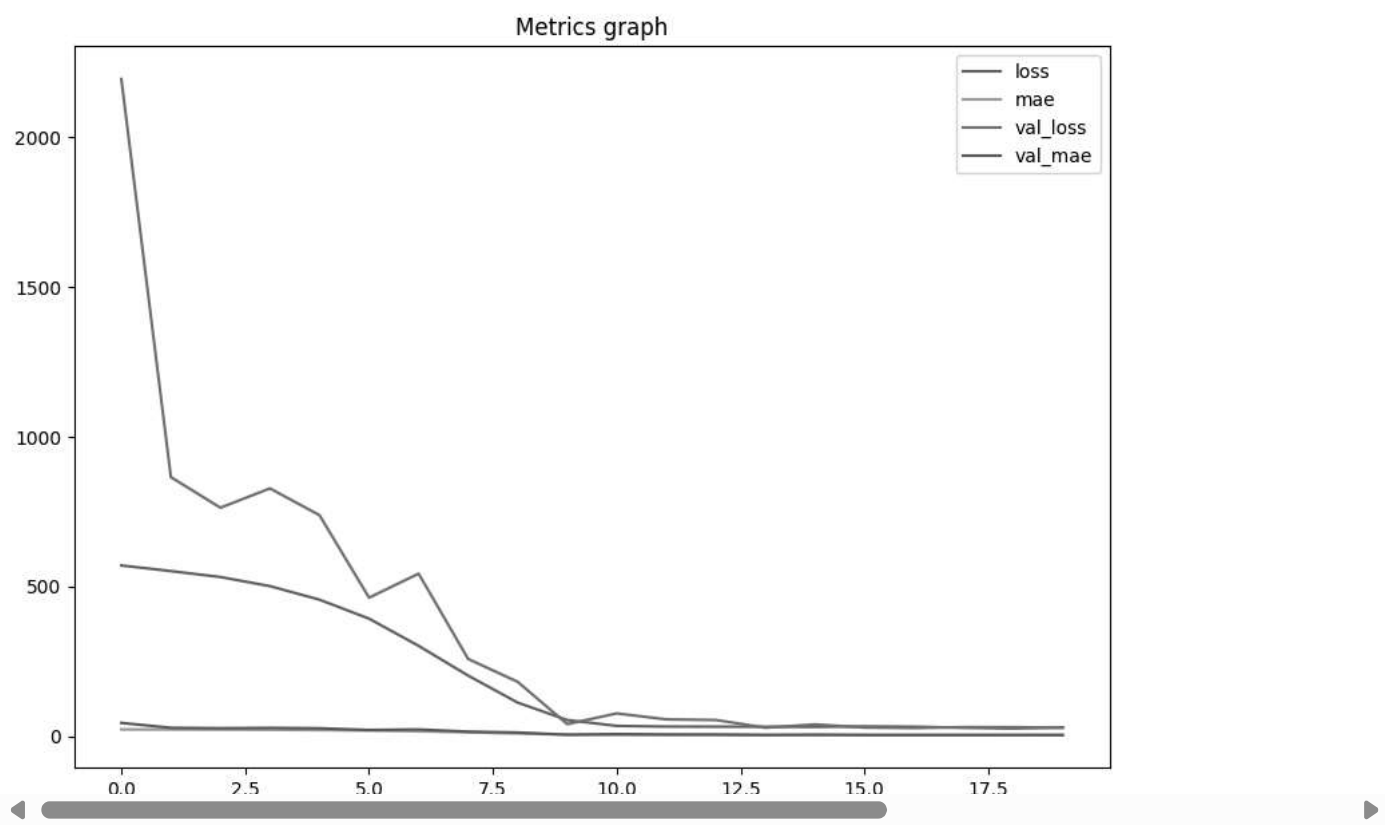
Total params: 6,901 (26.96 KB)

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

```
history = model.fit(x_train, y_train, batch_size=32, epochs=20, validation_data=(x_test, y_test))
```

```
Epoch 1/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 5s 69ms/step - loss: 566.2610 - mae: 22.1866 - val_loss: 2195.4170 - val_mae: 44.1316
Epoch 2/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 65ms/step - loss: 545.0226 - mae: 22.0348 - val_loss: 865.3286 - val_mae: 27.8704
Epoch 3/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 58ms/step - loss: 518.8692 - mae: 21.3549 - val_loss: 763.2159 - val_mae: 26.4279
Epoch 4/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 52ms/step - loss: 507.6797 - mae: 21.1946 - val_loss: 827.6428 - val_mae: 27.5940
Epoch 5/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 473.5003 - mae: 20.6118 - val_loss: 738.0662 - val_mae: 26.1877
Epoch 6/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 43ms/step - loss: 440.9617 - mae: 19.6571 - val_loss: 462.5768 - val_mae: 20.6089
Epoch 7/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 43ms/step - loss: 308.8321 - mae: 16.4774 - val_loss: 542.9050 - val_mae: 22.3664
Epoch 8/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - loss: 204.8676 - mae: 12.7731 - val_loss: 257.8780 - val_mae: 14.9512
Epoch 9/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - loss: 132.8249 - mae: 9.7740 - val_loss: 181.3808 - val_mae: 12.1404
Epoch 10/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - loss: 62.2791 - mae: 5.9631 - val_loss: 40.1941 - val_mae: 4.6248
Epoch 11/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - loss: 34.5409 - mae: 4.3488 - val_loss: 75.9025 - val_mae: 7.0207
Epoch 12/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - loss: 32.0851 - mae: 4.3401 - val_loss: 56.0865 - val_mae: 5.6735
Epoch 13/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - loss: 30.2474 - mae: 4.2756 - val_loss: 53.9263 - val_mae: 5.4860
Epoch 14/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - loss: 34.8608 - mae: 4.3029 - val_loss: 28.3176 - val_mae: 3.8497
Epoch 15/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - loss: 27.4388 - mae: 4.0282 - val_loss: 39.1400 - val_mae: 4.7373
Epoch 16/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - loss: 33.1061 - mae: 4.2458 - val_loss: 28.7656 - val_mae: 3.8003
Epoch 17/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - loss: 33.4246 - mae: 4.1804 - val_loss: 27.1520 - val_mae: 3.6834
Epoch 18/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - loss: 26.4472 - mae: 3.7951 - val_loss: 30.1213 - val_mae: 3.7754
Epoch 19/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - loss: 24.9190 - mae: 3.6405 - val_loss: 29.9513 - val_mae: 3.9915
Epoch 20/20
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - loss: 28.9134 - mae: 3.9870 - val_loss: 27.9116 - val_mae: 3.7991
```

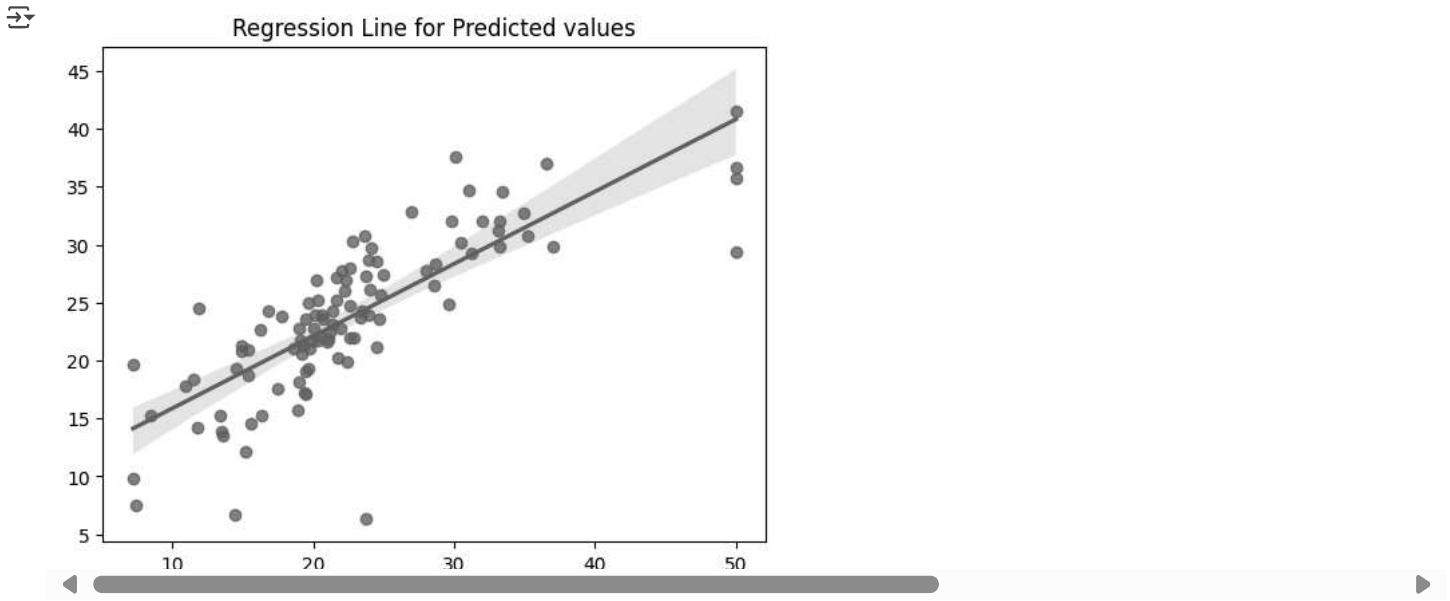
```
pd.DataFrame(history.history).plot(figsize=(10,7))
plt.title("Metrics graph")
plt.show()
```



```
y_pred = model.predict(x_test)
```

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 83ms/step
```

```
sns.regplot(x=y_test, y=y_pred)
plt.title("Regression Line for Predicted values")
plt.show()
```




```
def regression_metrics_display(y_test, y_pred):
  print(f"MAE is {metrics.mean_absolute_error(y_test, y_pred)}")
  print(f"MSE is {metrics.mean_squared_error(y_test,y_pred)}")
  print(f"R2 score is {metrics.r2_score(y_test, y_pred)}")
```

```
regression_metrics_display(y_test, y_pred)
```

```
MAE is 3.7990686248330507
MSE is 27.91156142341494
R2 score is 0.60799476381759
```

Start coding or generate with AI.