In [ ]:

```
# imports and definitions
import numpy as np
import pandas as pd
import tensorflow as tf
from IPython.display import display, HTML

# import logs
print(f'tensorflow version: {tf.__version__}')
print(f'num of GPU available: {len(tf.config.list_physical_devices())}')
```

```
tensorflow version: 2.10.0
num of GPU available: 1
```

In [ ]:

```
original_df = pd.read_csv('datasets/new_data.csv')
df = original_df
print(f'working dataset:')
display(df.head(5))
```

```
working dataset:
```

	Dp	Tt	V	q	Tt/Dp	Dmi/Dp	Dmj/Dp	Dmi	Dmj	Vh/Vp
0	6	6	2	35	1.0	1.67	2.17	10.02	13.02	5.42
1	6	6	3	50	1.0	2.17	2.83	13.02	16.98	9.21
2	6	6	4	60	1.0	2.33	3.00	13.98	18.00	10.50
3	6	6	2	70	1.0	0.00	0.00	0.00	0.00	0.00
4	6	6	3	35	1.0	2.33	2.67	13.98	16.02	9.33

In [ ]:

```
# checking of nulls / na
a = pd.DataFrame(df.isnull().sum())
a['# of null values'] = a[0]
null_val = a[['# of null values']]
print(f'Before dropping null values:\n# of Rows, Columns: {df.shape}')
display(null_val)

# removing nulls / na
df = df.dropna(axis=0)
a = pd.DataFrame(df.isnull().sum())
a['# of null values'] = a[0]
null_val = a[['# of null values']]
print(f'After dropping null values:\n# of Rows, Columns: {df.shape}')
display(null_val)
```

```
Before dropping null values:
# of Rows, Columns: (36, 10)
```

	# of null values
Dp	0
Tt	0
V	0
q	0
Tt/Dp	0
Dmi/Dp	0
Dmj/Dp	0
Dmi	0
Dmj	0
Vh/Vp	0

```
After dropping null values:
# of Rows, Columns: (36, 10)
```

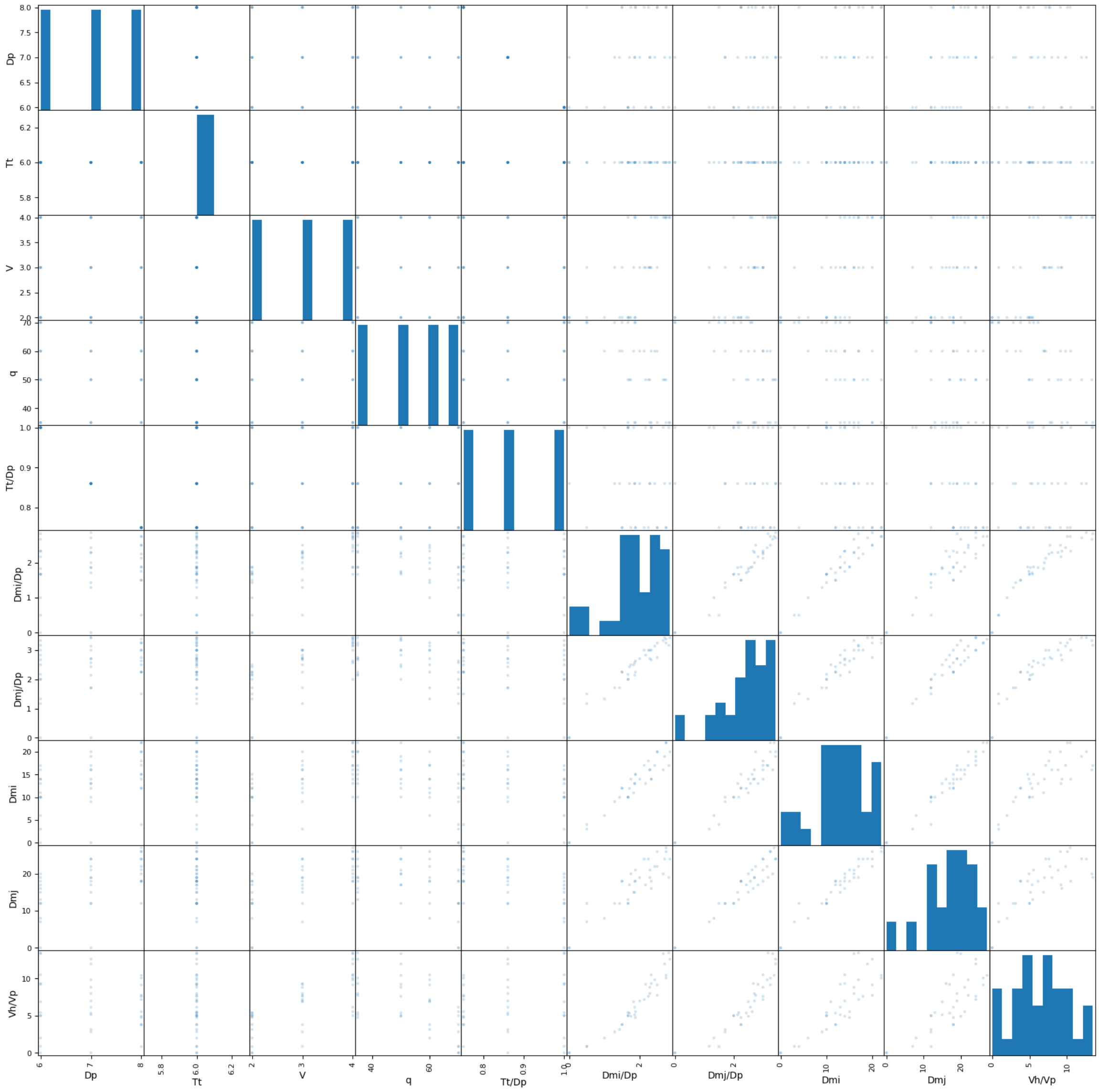
	# of null values
Dp	0
Tt	0
V	0
q	0
Tt/Dp	0
Dmi/Dp	0
Dmj/Dp	0
Dmi	0
Dmj	0
Vh/Vp	0

In [ ]:

```
c = pd.plotting.scatter_matrix(df, alpha=0.2, figsize=(20, 20), diagonal=
c
```

```
/home/crimson/miniconda3/envs/tf/lib/python3.9/site-packages/pandas/plott
ing/_matplotlib/misc.py:101: UserWarning: Attempting to set identical low
and high xlims makes transformation singular; automatically expanding.
  ax.set_xlim(boundaries_list[j])
/home/crimson/miniconda3/envs/tf/lib/python3.9/site-packages/pandas/plott
ing/_matplotlib/misc.py:102: UserWarning: Attempting to set identical low
and high ylims makes transformation singular; automatically expanding.
  ax.set_ylim(boundaries_list[i])
/home/crimson/miniconda3/envs/tf/lib/python3.9/site-packages/pandas/plott
ing/_matplotlib/misc.py:92: UserWarning: Attempting to set identical low
and high xlims makes transformation singular; automatically expanding.
  ax.set_xlim(boundaries_list[i])
```

```
Out[ ]: array([[<AxesSubplot: xlabel='Dp', ylabel='Dp'>,
                <AxesSubplot: xlabel='Tt', ylabel='Dp'>,
                <AxesSubplot: xlabel='V', ylabel='Dp'>,
                <AxesSubplot: xlabel='q', ylabel='Dp'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Dp'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Dp'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Dp'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Dp'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Dp'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Dp'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Tt'>,
                <AxesSubplot: xlabel='Tt', ylabel='Tt'>,
                <AxesSubplot: xlabel='V', ylabel='Tt'>,
                <AxesSubplot: xlabel='q', ylabel='Tt'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Tt'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Tt'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Tt'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Tt'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Tt'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Tt'>],
               [<AxesSubplot: xlabel='Dp', ylabel='V'>,
                <AxesSubplot: xlabel='Tt', ylabel='V'>,
                <AxesSubplot: xlabel='V', ylabel='V'>,
                <AxesSubplot: xlabel='q', ylabel='V'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='V'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='V'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='V'>,
                <AxesSubplot: xlabel='Dmi', ylabel='V'>,
                <AxesSubplot: xlabel='Dmj', ylabel='V'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='V'>],
               [<AxesSubplot: xlabel='Dp', ylabel='q'>,
                <AxesSubplot: xlabel='Tt', ylabel='q'>,
                <AxesSubplot: xlabel='V', ylabel='q'>,
                <AxesSubplot: xlabel='q', ylabel='q'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='q'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='q'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='q'>,
                <AxesSubplot: xlabel='Dmi', ylabel='q'>,
                <AxesSubplot: xlabel='Dmj', ylabel='q'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='q'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Tt', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='V', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='q', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Tt/Dp'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Tt/Dp'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Tt', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='V', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='q', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Dmi/Dp'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Dmi/Dp'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Tt', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='V', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='q', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Dmj/Dp'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Dmj/Dp'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Tt', ylabel='Dmi'>,
                <AxesSubplot: xlabel='V', ylabel='Dmi'>,
                <AxesSubplot: xlabel='q', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Dmi'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Dmi'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Tt', ylabel='Dmj'>,
                <AxesSubplot: xlabel='V', ylabel='Dmj'>,
                <AxesSubplot: xlabel='q', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Dmj'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Dmj'>],
               [<AxesSubplot: xlabel='Dp', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Tt', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='V', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='q', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Tt/Dp', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Dmi/Dp', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Dmj/Dp', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Dmi', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Dmj', ylabel='Vh/Vp'>,
                <AxesSubplot: xlabel='Vh/Vp', ylabel='Vh/Vp'>]], dtype=object)
```



In [ ]:

```
# feature engineering

display(df.head(10))
```

	Dp	Tt	V	q	Tt/Dp	Dmi/Dp	Dmj/Dp	Dmi	Dmj	Vh/Vp
0	6	6	2	35	1.0	1.67	2.17	10.02	13.02	5.42
1	6	6	3	50	1.0	2.17	2.83	13.02	16.98	9.21
2	6	6	4	60	1.0	2.33	3.00	13.98	18.00	10.50
3	6	6	2	70	1.0	0.00	0.00	0.00	0.00	0.00
4	6	6	3	35	1.0	2.33	2.67	13.98	16.02	9.33
5	6	6	4	50	1.0	2.67	3.33	16.02	19.98	13.34
6	6	6	2	60	1.0	1.00	1.33	6.00	7.98	2.00
7	6	6	3	70	1.0	0.50	1.17	3.00	7.02	0.88
8	6	6	4	35	1.0	2.83	3.17	16.98	19.02	13.46
9	6	6	2	50	1.0	1.67	2.00	10.02	12.00	5.00

In [ ]:

```
# Split into train and eval
SCALE = 10
BATCH_SIZE=2
SEED = 1024

traindf = df.sample(frac=0.9, random_state=SEED)
evaldf = df.drop(traindf.index)

print('# of Rows, Columns: ',df.shape)
display(df.head(10))

print(f'traindf shape: {traindf.shape}, evaldf shape: {evaldf.shape}')
```

```
# of Rows, Columns:  (36, 10)
```

	Dp	Tt	V	q	Tt/Dp	Dmi/Dp	Dmj/Dp	Dmi	Dmj	Vh/Vp
0	6	6	2	35	1.0	1.67	2.17	10.02	13.02	5.42
1	6	6	3	50	1.0	2.17	2.83	13.02	16.98	9.21
2	6	6	4	60	1.0	2.33	3.00	13.98	18.00	10.50
3	6	6	2	70	1.0	0.00	0.00	0.00	0.00	0.00
4	6	6	3	35	1.0	2.33	2.67	13.98	16.02	9.33
5	6	6	4	50	1.0	2.67	3.33	16.02	19.98	13.34
6	6	6	2	60	1.0	1.00	1.33	6.00	7.98	2.00
7	6	6	3	70	1.0	0.50	1.17	3.00	7.02	0.88
8	6	6	4	35	1.0	2.83	3.17	16.98	19.02	13.46
9	6	6	2	50	1.0	1.67	2.00	10.02	12.00	5.00

```
traindf shape: (32, 10), evaldf shape: (4, 10)
```

In [ ]:

```
n_neurons = 16
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(n_neurons, input_dim=9, activation='relu'),
    tf.keras.layers.Dense(n_neurons, activation='relu'),
    tf.keras.layers.Dense(n_neurons, activation='relu'),
    tf.keras.layers.Dense(1),
])

model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_24 (Dense)            (None, 16)                160

 dense_25 (Dense)            (None, 16)                272

 dense_26 (Dense)            (None, 16)                272

 dense_27 (Dense)            (None, 1)                 17

=================================================================
Total params: 721
Trainable params: 721
Non-trainable params: 0
_________________________________________________________________
```

In [ ]:

```python
loss = 'mean_squared_logarithmic_error'
opt = 'Adam'
model.compile(
    loss=loss,
    # optimizer=tf.keras.optimizers.SGD(learning_rate=1e-8),
    optimizer=opt,
    metrics=[
        tf.keras.metrics.MeanSquaredError(),
        # tf.keras.metrics.Accuracy(),
    ]
)
```

In [ ]:

```python
train_labels = traindf.pop('Vh/Vp')
eval_labels = evaldf.pop('Vh/Vp')
# display(train_labels.head(10))
# display(eval_labels.head(10))
```

In [ ]:

```python
%%time
import tensorboard
import datetime
log_dir = f'logs/{str(n_neurons)}-{str(opt)}-{str(loss)}'
tensorboard = tf.keras.callbacks.TensorBoard(log_dir=log_dir)

history = model.fit(
    x = traindf, y = train_labels,
    epochs=50,
    validation_split=0.1,
    batch_size=BATCH_SIZE,
    callbacks=[tensorboard]
)
```

```
Epoch 1/50
```

```
/home/crimson/miniconda3/envs/tf/lib/python3.9/site-packages/keras/engine
/data_adapter.py:1699: FutureWarning: The behavior of `series[i:j]` with
an integer-dtype index is deprecated. In a future version, this will be t
reated as *label-based* indexing, consistent with e.g. `series[i]` lookup
s. To retain the old behavior, use `series.iloc[i:j]`. To get the future
behavior, use `series.loc[i:j]`.
  return t[start:end]
```

```
14/14 [==============================] - 1s 13ms/step - loss: 2.2860 - me
an_squared_error: 49.2726 - val_loss: 1.0487 - val_mean_squared_error: 1
9.0385
Epoch 2/50
14/14 [==============================] - 0s 4ms/step - loss: 1.1571 - mea
n_squared_error: 36.5310 - val_loss: 0.8167 - val_mean_squared_error: 14.
1515
Epoch 3/50
14/14 [==============================] - 0s 4ms/step - loss: 0.7607 - mea
n_squared_error: 29.0253 - val_loss: 0.7163 - val_mean_squared_error: 10.
9981
Epoch 4/50
14/14 [==============================] - 0s 4ms/step - loss: 0.5803 - mea
n_squared_error: 23.1694 - val_loss: 0.7158 - val_mean_squared_error: 9.5
064
Epoch 5/50
14/14 [==============================] - 0s 4ms/step - loss: 0.4788 - mea
n_squared_error: 20.1221 - val_loss: 0.4812 - val_mean_squared_error: 6.7
254
Epoch 6/50
14/14 [==============================] - 0s 4ms/step - loss: 0.3927 - mea
n_squared_error: 17.7274 - val_loss: 0.1695 - val_mean_squared_error: 4.7
656
Epoch 7/50
14/14 [==============================] - 0s 5ms/step - loss: 0.2830 - mea
n_squared_error: 15.7722 - val_loss: 0.0795 - val_mean_squared_error: 4.1
398
Epoch 8/50
14/14 [==============================] - 0s 4ms/step - loss: 0.1986 - mea
n_squared_error: 13.6951 - val_loss: 0.0467 - val_mean_squared_error: 3.1
342
Epoch 9/50
14/14 [==============================] - 0s 4ms/step - loss: 0.1513 - mea
n_squared_error: 11.3754 - val_loss: 0.0292 - val_mean_squared_error: 2.4
596
Epoch 10/50
14/14 [==============================] - 0s 4ms/step - loss: 0.1187 - mea
n_squared_error: 9.6521 - val_loss: 0.0178 - val_mean_squared_error: 1.96
79
Epoch 11/50
14/14 [==============================] - 0s 5ms/step - loss: 0.1025 - mea
n_squared_error: 8.4537 - val_loss: 0.0204 - val_mean_squared_error: 1.27
35
Epoch 12/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0909 - mea
n_squared_error: 7.2602 - val_loss: 0.0080 - val_mean_squared_error: 1.35
96
Epoch 13/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0832 - mea
n_squared_error: 7.0879 - val_loss: 0.0121 - val_mean_squared_error: 0.94
24
Epoch 14/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0783 - mea
n_squared_error: 6.3250 - val_loss: 0.0034 - val_mean_squared_error: 1.22
72
Epoch 15/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0685 - mea
n_squared_error: 5.9320 - val_loss: 0.0069 - val_mean_squared_error: 0.89
32
Epoch 16/50
```

```
14/14 [==============================] - 0s 4ms/step - loss: 0.0792 - mea
n_squared_error: 5.5381 - val_loss: 0.0067 - val_mean_squared_error: 0.83
33
Epoch 17/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0686 - mea
n_squared_error: 5.7790 - val_loss: 0.0021 - val_mean_squared_error: 1.29
19
Epoch 18/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0697 - mea
n_squared_error: 5.3721 - val_loss: 0.0047 - val_mean_squared_error: 1.18
10
Epoch 19/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0582 - mea
n_squared_error: 4.8451 - val_loss: 0.0075 - val_mean_squared_error: 0.93
54
Epoch 20/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0536 - mea
n_squared_error: 4.7466 - val_loss: 0.0016 - val_mean_squared_error: 1.00
05
Epoch 21/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0524 - mea
n_squared_error: 4.4273 - val_loss: 9.0610e-04 - val_mean_squared_error:
1.0051
Epoch 22/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0524 - mea
n_squared_error: 4.4774 - val_loss: 0.0012 - val_mean_squared_error: 1.06
23
Epoch 23/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0530 - mea
n_squared_error: 4.3961 - val_loss: 0.0124 - val_mean_squared_error: 0.74
10
Epoch 24/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0570 - mea
n_squared_error: 4.1637 - val_loss: 4.8360e-04 - val_mean_squared_error:
1.2653
Epoch 25/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0537 - mea
n_squared_error: 4.3125 - val_loss: 0.0032 - val_mean_squared_error: 0.94
17
Epoch 26/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0470 - mea
n_squared_error: 4.0418 - val_loss: 0.0053 - val_mean_squared_error: 0.80
93
Epoch 27/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0486 - mea
n_squared_error: 3.9643 - val_loss: 5.2382e-04 - val_mean_squared_error:
0.9103
Epoch 28/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0441 - mea
n_squared_error: 3.9717 - val_loss: 0.0054 - val_mean_squared_error: 0.67
61
Epoch 29/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0557 - mea
n_squared_error: 3.8838 - val_loss: 0.0037 - val_mean_squared_error: 1.49
16
Epoch 30/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0436 - mea
n_squared_error: 3.8001 - val_loss: 0.0071 - val_mean_squared_error: 0.61
62
Epoch 31/50
```

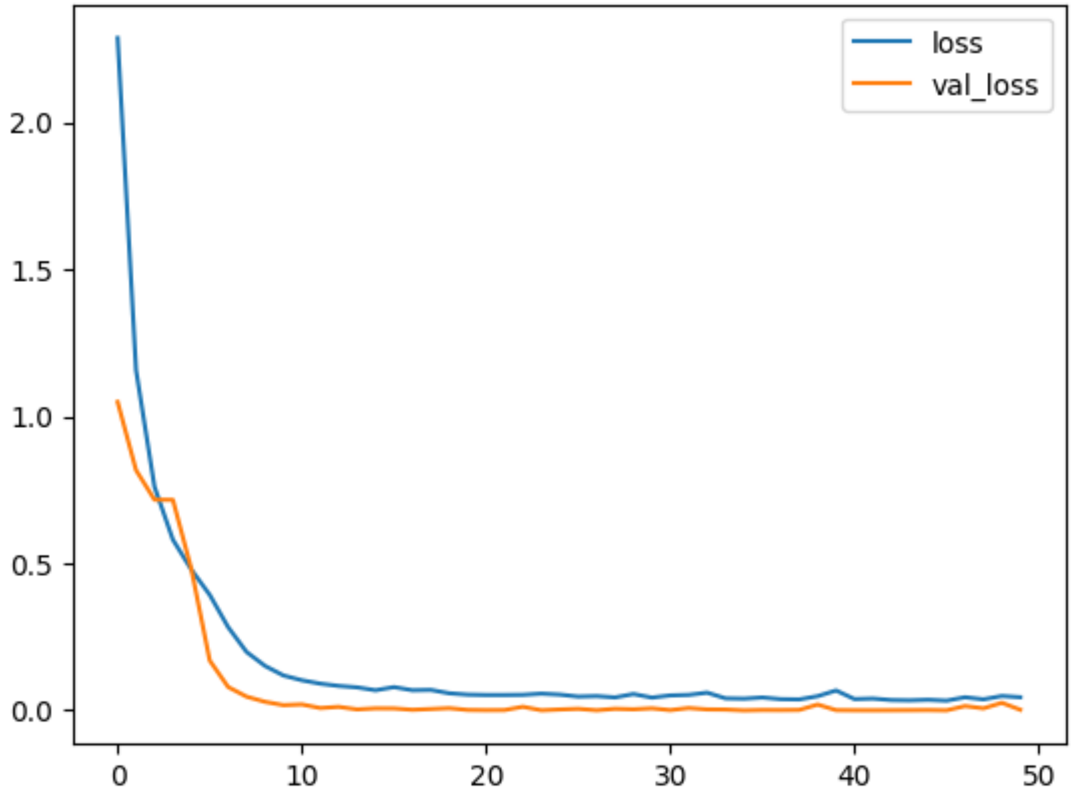
```
14/14 [==============================] - 0s 4ms/step - loss: 0.0509 - mea
n_squared_error: 3.6133 - val_loss: 9.4892e-04 - val_mean_squared_error:
1.3147
Epoch 32/50
14/14 [==============================] - 0s 7ms/step - loss: 0.0527 - mea
n_squared_error: 4.1920 - val_loss: 0.0081 - val_mean_squared_error: 0.88
03
Epoch 33/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0603 - mea
n_squared_error: 4.5292 - val_loss: 0.0034 - val_mean_squared_error: 1.41
77
Epoch 34/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0410 - mea
n_squared_error: 3.6192 - val_loss: 0.0028 - val_mean_squared_error: 0.82
33
Epoch 35/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0399 - mea
n_squared_error: 3.4958 - val_loss: 8.7470e-05 - val_mean_squared_error:
1.1736
Epoch 36/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0439 - mea
n_squared_error: 3.4699 - val_loss: 0.0013 - val_mean_squared_error: 0.84
03
Epoch 37/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0382 - mea
n_squared_error: 3.5529 - val_loss: 0.0011 - val_mean_squared_error: 0.78
47
Epoch 38/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0374 - mea
n_squared_error: 3.3260 - val_loss: 0.0016 - val_mean_squared_error: 0.75
97
Epoch 39/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0484 - mea
n_squared_error: 3.4507 - val_loss: 0.0199 - val_mean_squared_error: 0.85
63
Epoch 40/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0677 - mea
n_squared_error: 3.8328 - val_loss: 9.6616e-04 - val_mean_squared_error:
0.7543
Epoch 41/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0384 - mea
n_squared_error: 3.4567 - val_loss: 4.7281e-04 - val_mean_squared_error:
0.7886
Epoch 42/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0402 - mea
n_squared_error: 3.4895 - val_loss: 3.7000e-04 - val_mean_squared_error:
0.8650
Epoch 43/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0352 - mea
n_squared_error: 3.2459 - val_loss: 5.3935e-04 - val_mean_squared_error:
0.7946
Epoch 44/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0343 - mea
n_squared_error: 3.1029 - val_loss: 7.6509e-04 - val_mean_squared_error:
0.7450
Epoch 45/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0361 - mea
n_squared_error: 3.1723 - val_loss: 0.0012 - val_mean_squared_error: 0.85
74
Epoch 46/50
```

```
14/14 [==============================] - 0s 4ms/step - loss: 0.0332 - mea
n_squared_error: 3.0124 - val_loss: 2.8921e-04 - val_mean_squared_error:
1.0357
Epoch 47/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0447 - mea
n_squared_error: 3.2745 - val_loss: 0.0148 - val_mean_squared_error: 0.70
87
Epoch 48/50
14/14 [==============================] - 0s 5ms/step - loss: 0.0377 - mea
n_squared_error: 2.8558 - val_loss: 0.0070 - val_mean_squared_error: 1.79
59
Epoch 49/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0490 - mea
n_squared_error: 2.9762 - val_loss: 0.0258 - val_mean_squared_error: 1.08
23
Epoch 50/50
14/14 [==============================] - 0s 4ms/step - loss: 0.0447 - mea
n_squared_error: 2.8711 - val_loss: 0.0022 - val_mean_squared_error: 1.06
87
CPU times: user 4.15 s, sys: 328 ms, total: 4.48 s
Wall time: 3.69 s
```

In [ ]:

```python
import matplotlib.pyplot as plt

plt.plot(history.history['loss'], label='loss')
plt.plot(history.history['val_loss'], label='val_loss')
# plt.plot(history.history['mean_squared_error'], label='rmse')
plt.legend()
plt.show()
```



In [ ]:

```python
results = model.evaluate(x = evaldf, y = eval_labels)
print(f'test loss, test acc: {results}')
```

```
1/1 [==============================] - 0s 118ms/step - loss: 0.0213 - mea
n_squared_error: 1.6445
test loss, test acc: [0.02127729542553425, 1.644518256187439]
```