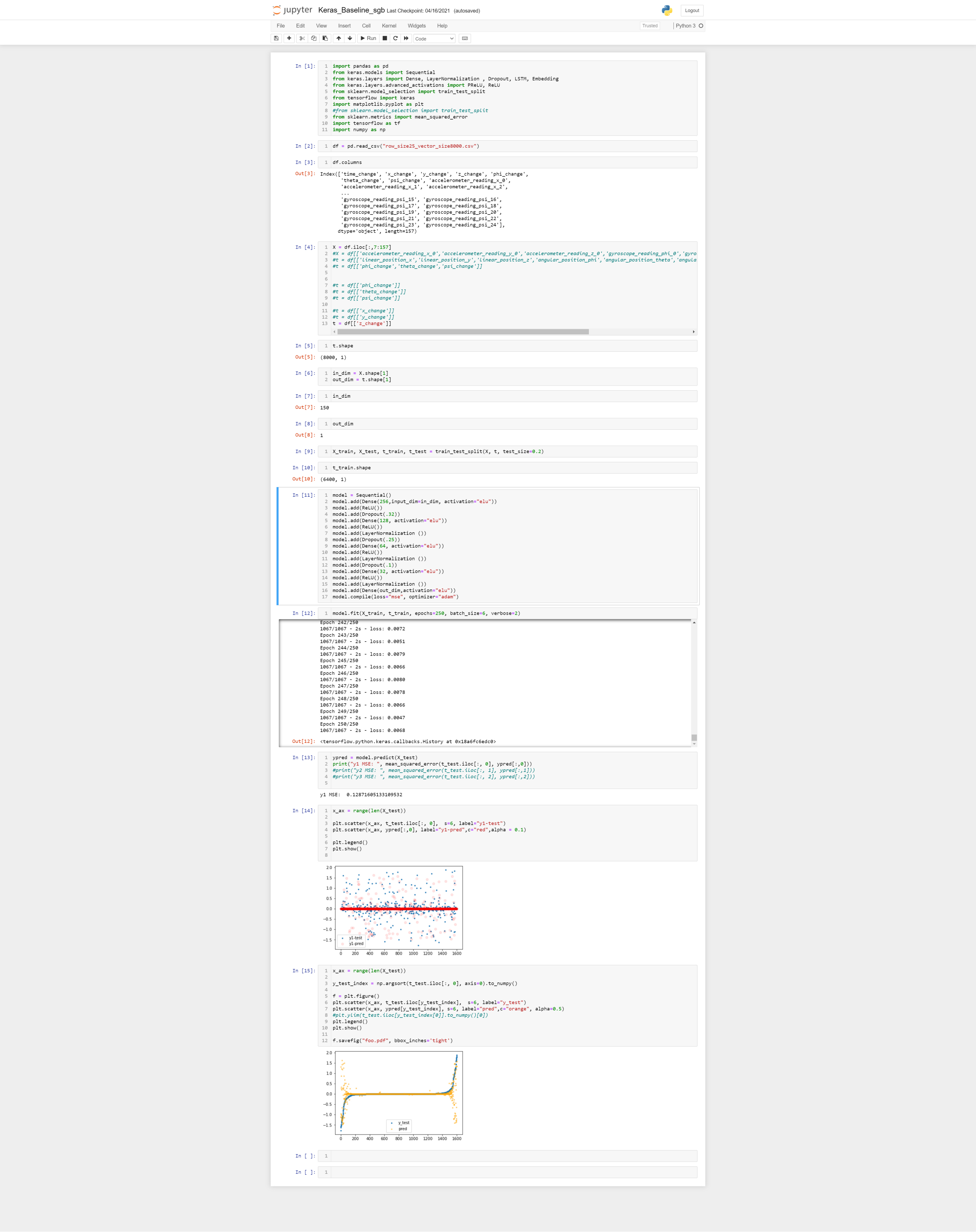

```
In [1]:
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense, LayerNormalization , Dropout, LSTM, Embedding
from keras.layers.advanced_activations import PReLU, ReLU
from sklearn.model_selection import train_test_split
from tensorflow import keras
import matplotlib.pyplot as plt
#from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import tensorflow as tf
import numpy as np
```

```
In [2]:
df = pd.read_csv("row_size25_vector_size8000.csv")
```

```
In [3]:
df.columns
```

```
Out[3]: Index(['time_change', 'x_change', 'y_change', 'z_change', 'phi_change',
               'theta_change', 'psi_change', 'accelerometer_reading_x_0',
               'accelerometer_reading_x_1', 'accelerometer_reading_x_2',
               ...
               'gyroscope_reading_psi_15', 'gyroscope_reading_psi_16',
               'gyroscope_reading_psi_17', 'gyroscope_reading_psi_18',
               'gyroscope_reading_psi_19', 'gyroscope_reading_psi_20',
               'gyroscope_reading_psi_21', 'gyroscope_reading_psi_22',
               'gyroscope_reading_psi_23', 'gyroscope_reading_psi_24'],
              dtype='object', length=157)
```

```
In [4]:
X = df.iloc[:,7:157]
#X = df[['accelerometer_reading_x_0','accelerometer_reading_y_0','accelerometer_reading_z_0','gyroscope_reading_phi_0','gyro
#t = df[['linear_position_x','linear_position_y','linear_position_z','angular_position_phi','angular_position_theta','angula
#t = df[['phi_change','theta_change','psi_change']]


#t = df[['phi_change']]
#t = df[['theta_change']]
#t = df[['psi_change']]

#t = df[['x_change']]
#t = df[['y_change']]
t = df[['z_change']]
```

```
In [5]:
t.shape
```

```
Out[5]: (8000, 1)
```

```
In [6]:
in_dim = X.shape[1]
out_dim = t.shape[1]
```

```
In [7]:
in_dim
```

```
Out[7]: 150
```

```
In [8]:
out_dim
```

```
Out[8]: 1
```

```
In [9]:
X_train, X_test, t_train, t_test = train_test_split(X, t, test_size=0.2)
```

```
In [10]:
t_train.shape
```

```
Out[10]: (6400, 1)
```

```
In [11]:
model = Sequential()
model.add(Dense(256,input_dim=in_dim, activation="elu"))
model.add(ReLU())
model.add(Dropout(.32))
model.add(Dense(128, activation="elu"))
model.add(ReLU())
model.add(LayerNormalization ())
model.add(Dropout(.25))
model.add(Dense(64, activation="elu"))
model.add(ReLU())
model.add(LayerNormalization ())
model.add(Dropout(.1))
model.add(Dense(32, activation="elu"))
model.add(ReLU())
model.add(LayerNormalization ())
model.add(Dense(out_dim,activation="elu"))
model.compile(loss="mse", optimizer="adam")
```

```
In [12]:
model.fit(X_train, t_train, epochs=250, batch_size=6, verbose=2)
```

```
Epoch 242/250
1067/1067 - 2s - loss: 0.0072
Epoch 243/250
1067/1067 - 2s - loss: 0.0051
Epoch 244/250
1067/1067 - 2s - loss: 0.0079
Epoch 245/250
1067/1067 - 2s - loss: 0.0066
Epoch 246/250
1067/1067 - 2s - loss: 0.0080
Epoch 247/250
1067/1067 - 2s - loss: 0.0078
Epoch 248/250
1067/1067 - 2s - loss: 0.0066
Epoch 249/250
1067/1067 - 2s - loss: 0.0047
Epoch 250/250
1067/1067 - 2s - loss: 0.0068

Out[12]: <tensorflow.python.keras.callbacks.History at 0x18a6fc6edc0>
```

```
In [13]:
ypred = model.predict(X_test)
print("y1 MSE: ", mean_squared_error(t_test.iloc[:, 0], ypred[:,0]))
#print("y2 MSE: ", mean_squared_error(t_test.iloc[:, 1], ypred[:,1]))
#print("y3 MSE: ", mean_squared_error(t_test.iloc[:, 2], ypred[:,2]))

```

```
y1 MSE:  0.12871605133109532
```

```
In [14]:
x_ax = range(len(X_test))

plt.scatter(x_ax, t_test.iloc[:, 0],  s=6, label="y1-test")
plt.scatter(x_ax, ypred[:,0], label="y1-pred",c="red",alpha = 0.1)

plt.legend()
plt.show()

```

```
In [15]:
x_ax = range(len(X_test))

y_test_index = np.argsort(t_test.iloc[:, 0], axis=0).to_numpy()

f = plt.figure()
plt.scatter(x_ax, t_test.iloc[y_test_index],  s=6, label="y_test")
plt.scatter(x_ax, ypred[y_test_index], s=6, label="pred",c="orange", alpha=0.5)
#plt.ylim(t_test.iloc[y_test_index[0]].to_numpy()[0])
plt.legend()
plt.show()

f.savefig("foo.pdf", bbox_inches='tight')
```

```
In [ ]:
```

```
In [ ]:
```