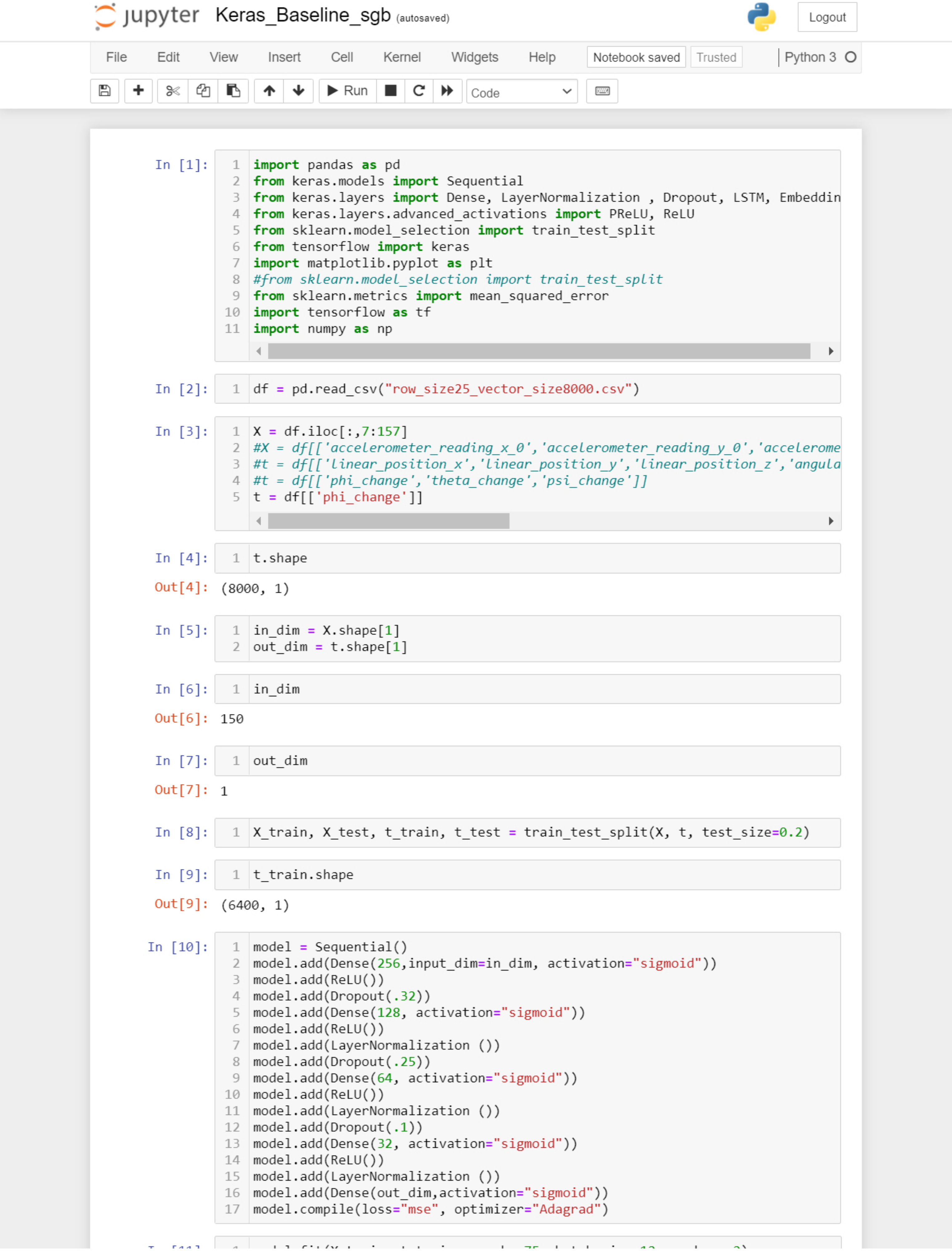


```
In [1]:  import pandas as pd
         from keras.models import Sequential
         from keras.layers import Dense, LayerNormalization , Dropout, LSTM, Embeddin
         from keras.layers.advanced_activations import PReLU, ReLU
         from sklearn.model_selection import train_test_split
         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]:  X = df.iloc[:,7:157]
         #X = df[['accelerometer_reading_x_0','accelerometer_reading_y_0','accelerome
         #t = df[['linear_position_x','linear_position_y','linear_position_z','angula
         #t = df[['phi_change','theta_change','psi_change']]
         t = df[['phi_change']]
```

```
In [4]:  t.shape

Out[4]:  (8000, 1)
```

```
In [5]:  in_dim = X.shape[1]
         out_dim = t.shape[1]
```

```
In [6]:  in_dim

Out[6]:  150
```

```
In [7]:  out_dim

Out[7]:  1
```

```
In [8]:  X_train, X_test, t_train, t_test = train_test_split(X, t, test_size=0.2)
```

```
In [9]:  t_train.shape

Out[9]:  (6400, 1)
```

```
In [10]: model = Sequential()
         model.add(Dense(256,input_dim=in_dim, activation="sigmoid"))
         model.add(ReLU())
         model.add(Dropout(.32))
         model.add(Dense(128, activation="sigmoid"))
         model.add(ReLU())
         model.add(LayerNormalization ())
         model.add(Dropout(.25))
         model.add(Dense(64, activation="sigmoid"))
         model.add(ReLU())
         model.add(LayerNormalization ())
         model.add(Dropout(.1))
         model.add(Dense(32, activation="sigmoid"))
         model.add(ReLU())
         model.add(LayerNormalization ())
         model.add(Dense(out_dim,activation="sigmoid"))
         model.compile(loss="mse", optimizer="Adagrad")
```

```
In [11]: model.fit(X_train, t_train, epochs=75, batch_size=12, verbose=2)
```

In [11]:
```
model.fit(X_train, t_train, epochs=75, batch_size=12, verbose=2)
```

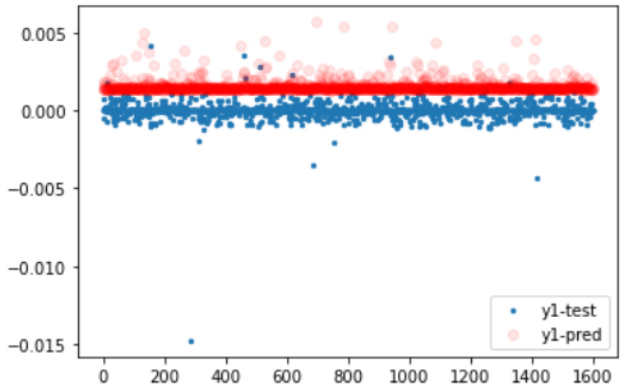
```
534/534 - 1s - loss: 3.6982e-05
Epoch 67/75
534/534 - 1s - loss: 4.1225e-05
Epoch 68/75
534/534 - 1s - loss: 3.5766e-05
Epoch 69/75
534/534 - 1s - loss: 3.9038e-05
Epoch 70/75
534/534 - 1s - loss: 4.0308e-05
Epoch 71/75
534/534 - 1s - loss: 3.7646e-05
Epoch 72/75
534/534 - 1s - loss: 3.9055e-05
Epoch 73/75
534/534 - 1s - loss: 3.5330e-05
Epoch 74/75
534/534 - 1s - loss: 3.8062e-05
Epoch 75/75
534/534 - 2s - loss: 3.7622e-05
```

Out[11]: `<tensorflow.python.keras.callbacks.History at 0x25968f6d580>`

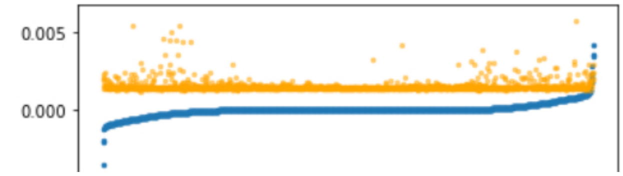
In [12]:
```
ypred = model.predict(X_test)
print("y1 MSE: ", mean_squared_error(t_test.iloc[:, 0], ypred[:,0]))
#print("y2 MSE: ", mean_squared_error(t_test.iloc[:, 1], ypred[:,1]))
#print("y3 MSE: ", mean_squared_error(t_test.iloc[:, 2], ypred[:,2]))

```

```
y1 MSE:  2.79816187961012e-06
```

In [13]:
```
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 [14]:
```
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
#plt.ylim(t_test.iloc[y_test_index[0]].to_numpy()[0])
plt.legend()
plt.show()

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



−0.005
−0.010
−0.015
0
200
400
600
800
1000
1200
1400
1600
y_test
pred

```
In [ ]:
```

```
In [ ]:
```