Keras_Baseline_sgb.ipynb

File Edit View Insert Runtime Tools Help All c...

Comment Share S

+ Code + Text RAM Disk Editing

```
[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
```

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

```
[3] X = df.iloc[:,7:157]
    #X = df[['accelerometer_reading_x_0','accelerometer_reading_y_0','accelerometer_readi
    #t = df[['linear_position_x','linear_position_y','linear_position_z','angular_positio
    #t = df[['phi_change','theta_change','psi_change']]
    t = df[['phi_change']]
```

```
[4] t.shape
```

```
(8000, 1)
```

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

```
[6] in_dim
```

```
150
```

```
[7] out_dim
```

```
1
```

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

```
[9] t_train.shape
```

```
(6400, 1)
```

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

```
[11] model.fit(X_train, t_train, epochs=75, batch_size=12, verbose=2)
     Epoch 47/75
     534/534 - 1s - loss: 4.8172e-05
     Epoch 48/75
     534/534 - 1s - loss: 2.1552e-05
     Epoch 49/75
     534/534 - 1s - loss: 2.1983e-05
     Epoch 50/75
     534/534 - 1s - loss: 4.1985e-05
     Epoch 51/75
     534/534 - 1s - loss: 2.2083e-05
     Epoch 52/75
     534/534 - 1s - loss: 5.3371e-05
     Epoch 53/75
     534/534 - 1s - loss: 1.5593e-05
     Epoch 54/75
     534/534 - 1s - loss: 1.9320e-05
     Epoch 55/75
     534/534 - 1s - loss: 2.2058e-05
     Epoch 56/75
     534/534 - 1s - loss: 6.7223e-05
     Epoch 57/75
     534/534 - 1s - loss: 2.8528e-05
     Epoch 58/75
     534/534 - 1s - loss: 4.2720e-05
     Epoch 59/75
     534/534 - 1s - loss: 1.1762e-05
     Epoch 60/75
     534/534 - 1s - loss: 6.1243e-05
     Epoch 61/75
     534/534 - 1s - loss: 1.6603e-05
     Epoch 62/75
     534/534 - 1s - loss: 1.1567e-05
     Epoch 63/75
     534/534 - 1s - loss: 2.6696e-05
     Epoch 64/75
     534/534 - 1s - loss: 1.5119e-05
     Epoch 65/75
```

```
534/534 - 1s - loss: 2.3085e-05
Epoch 66/75
534/534 - 1s - loss: 6.6132e-06
Epoch 67/75
534/534 - 1s - loss: 2.5537e-05
Epoch 68/75
534/534 - 1s - loss: 1.5744e-05
Epoch 69/75
534/534 - 1s - loss: 3.5975e-05
Epoch 70/75
534/534 - 1s - loss: 1.3149e-04
Epoch 71/75
534/534 - 1s - loss: 1.6317e-05
Epoch 72/75
534/534 - 1s - loss: 1.4555e-05
Epoch 73/75
534/534 - 1s - loss: 7.0691e-06
Epoch 74/75
534/534 - 1s - loss: 2.5063e-05
Epoch 75/75
534/534 - 1s - loss: 3.5924e-05
<tensorflow.python.keras.callbacks.History at 0x7f5380078650>
```

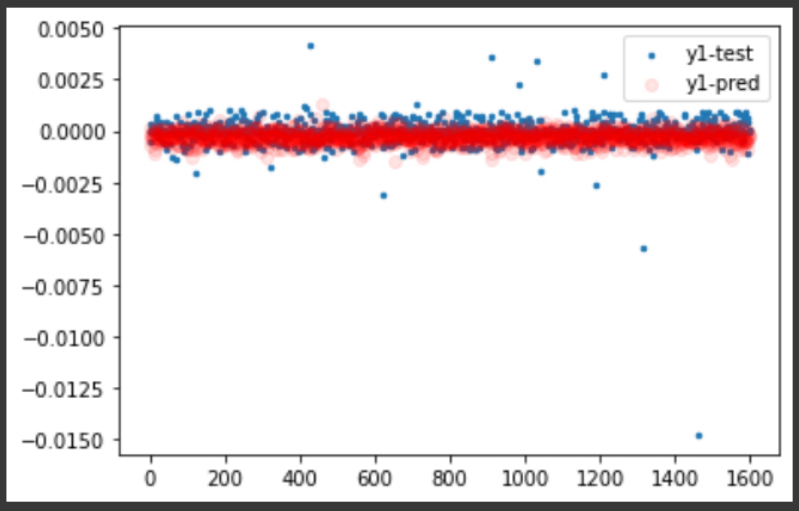
[12]
```
ypred = model.predict(X_test)
print("y1 MSE: ", mean_squared_error(t_test.iloc[:, 0], ypred[:,0]))
#print("y2 MSE: ", mean_squared_error(t_test.iloc[:, 1], ypred[:,1]))
#print("y3 MSE: ", mean_squared_error(t_test.iloc[:, 2], ypred[:,2]))
```

```
y1 MSE:  4.806872898468306e-07
```

[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()
```

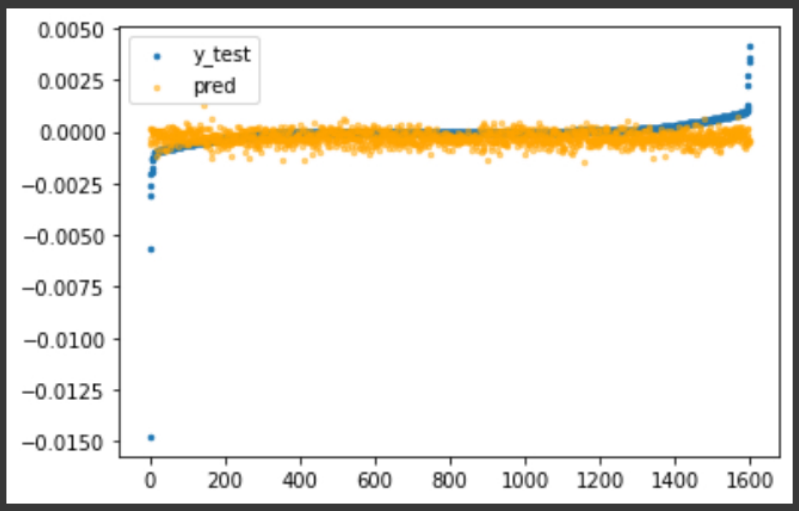



```
x_ax = range(len(X_test))
```

```
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')
```




```
[ ]
```

✓ 0s completed at 8:28 PM