

Task 8.1: Forecasting model

Data pre-processing:

```
from sklearn import preprocessing
le=preprocessing.LabelEncoder()
train_x.iloc[:,1]=le.fit_transform(train_x.iloc[:,1].values)
```

```
scaler = MinMaxScaler()
```

```
train_x.iloc[:,1]=scaler.fit_transform(train_x.iloc[:,1:].values)
```

```
C:\Users\sagar\anaconda3\lib\site-packages\sklearn\utils\validation.py:590: DataConversionWarning: Data with input dtype int64
was converted to float64 by MinMaxScaler.
  warnings.warn(msg, DataConversionWarning)
```

```
train_size = int(len(dataset) * 0.67)
test_size = len(dataset) - train_size
print(train_size)
```

```
30897
```

```
test_size = len(dataset) - train_size
print(test_size)
```

SimpleRNN:

```
reg.add(SimpleRNN(units=100, input_shape=(x_train.shape[1],x_train.shape[2]),name='simple_rnn'))
reg.add(Dropout(0.5,name = 'drp1'))
reg.add(Dense(units=10,name = 'fc1'))
reg.add(Dropout(0.2,name = 'drp2'))
reg.add(Dense(units=1,name='output'))

adm=Adam(lr=0.0001)
reg.compile(optimizer=adm, loss='mae')
reg.summary()
model=reg.fit(x_train,y_train,epochs=30,batch_size=100,validation_split=0.20)
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
simple_rnn (SimpleRNN)	(None, 100)	10300
drp1 (Dropout)	(None, 100)	0
fc1 (Dense)	(None, 10)	1010
drp2 (Dropout)	(None, 10)	0
output (Dense)	(None, 1)	11
Total params: 11,321		
Trainable params: 11,321		
Non-trainable params: 0		

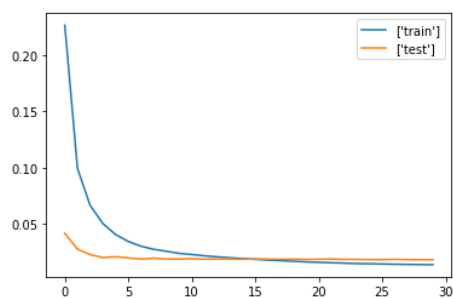
WARNING:tensorflow:From C:\Users\sagar\anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.

Train on 36868 samples, validate on 9218 samples

```
Epoch 1/30
36868/36868 [=====] - 9s 248us/step - loss: 0.2270 - val_loss: 0.0414
Epoch 2/30
36868/36868 [=====] - 9s 231us/step - loss: 0.0992 - val_loss: 0.0272
Epoch 3/30
36868/36868 [=====] - 9s 231us/step - loss: 0.0662 - val_loss: 0.0223
Epoch 4/30
36868/36868 [=====] - 9s 231us/step - loss: 0.0500 - val_loss: 0.0198
Epoch 5/30
36868/36868 [=====] - 9s 233us/step - loss: 0.0403 - val_loss: 0.0204
```

```
pred=reg.predict(x_train)
err=np.mean(np.abs(y_train-pred))
print("MAE error for standard averaging:",err)
```

MAE error for standard averaging: 0.013902409068076065



```
pred=reg.predict(x_train)
err=np.mean(np.abs(y_train-pred))
print("MAE error for standard averaging:",err)
```

MAE error for standard averaging: 0.013902409068076065

LSTM:

```
reg=Sequential()
print("x=",x_train.shape[1])
reg.add(LSTM(units=100, input_shape=(x_train.shape[1],x_train.shape[2]),name='LSTM'))
reg.add(Dropout(0.5,name = 'drp1'))
reg.add(Dense(units=10,name = 'fc1'))
reg.add(Dropout(0.2,name = 'drp2'))
reg.add(Dense(units=1,name='fc2'))
adm=Adam(lr=0.0001)
reg.compile(optimizer=adm, loss='mae')
reg.summary()
model=reg.fit(x_train,y_train,epochs=30,batch_size=100,validation_split=0.20)
```

x= 30

Model: "sequential_2"

Layer (type)	Output Shape	Param #
LSTM (LSTM)	(None, 100)	41200
drp1 (Dropout)	(None, 100)	0
fc1 (Dense)	(None, 10)	1010
drp2 (Dropout)	(None, 10)	0
fc2 (Dense)	(None, 1)	11

=====

Total params: 42,221

Trainable params: 42,221

Non-trainable params: 0

=====

Train on 36868 samples, validate on 9218 samples

Epoch 1/30
36868/36868 [=====] - 13s 347us/step - loss: 0.0422 - val_loss: 0.0182

Epoch 2/30
36868/36868 [=====] - 12s 337us/step - loss: 0.0225 - val_loss: 0.0194

Epoch 3/30
36868/36868 [=====] - 12s 335us/step - loss: 0.0188 - val_loss: 0.0189

Epoch 4/30
36868/36868 [=====] - 12s 335us/step - loss: 0.0169 - val_loss: 0.0190

Epoch 5/30
36868/36868 [=====] - 12s 337us/step - loss: 0.0157 - val_loss: 0.0179

Epoch 6/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0140 - val_loss: 0.0161

Epoch 7/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0129 - val_loss: 0.0154

Epoch 8/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0120 - val_loss: 0.0150

Epoch 9/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0114 - val_loss: 0.0148

Epoch 10/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0109 - val_loss: 0.0147

Epoch 11/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0105 - val_loss: 0.0146

Epoch 12/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0102 - val_loss: 0.0145

Epoch 13/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0100 - val_loss: 0.0144

Epoch 14/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0098 - val_loss: 0.0143

Epoch 15/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0096 - val_loss: 0.0142

Epoch 16/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0095 - val_loss: 0.0141

Epoch 17/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0094 - val_loss: 0.0140

Epoch 18/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0093 - val_loss: 0.0140

Epoch 19/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0092 - val_loss: 0.0140

Epoch 20/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0091 - val_loss: 0.0140

Epoch 21/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 22/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 23/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 24/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 25/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 26/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 27/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 28/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 29/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

Epoch 30/30
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

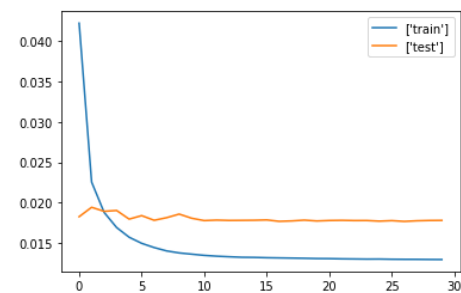
36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140

36868/36868 [=====] - 12s 336us/step - loss: 0.0090 - val_loss: 0.0140



```
pred=reg.predict(x_train)
err=np.mean(np.abs(y_train-pred))
print("MAE error for standard averaging:",err)
```

MAE error for standard averaging: 0.013870752687459686

GRU:

```
reg=Sequential()
print("x=",x_train.shape[1])
reg.add(GRU(units=100, input_shape=(x_train.shape[1],x_train.shape[2]),name='GRU'))
reg.add(Dropout(0.5,name = 'drp1'))
reg.add(Dense(units=10,name = 'fc1'))
reg.add(Dropout(0.2,name = 'drp2'))
reg.add(Dense(units=1,name='fc2'))
adm=Adam(lr=0.0001)
reg.compile(optimizer=adm, loss='mae')
reg.summary()
model=reg.fit(x_train,y_train,epochs=30,batch_size=100,validation_split=0.20)
```

x= 30

Model: "sequential_3"

Layer (type)	Output Shape	Param #
GRU (GRU)	(None, 100)	30900
drp1 (Dropout)	(None, 100)	0
fc1 (Dense)	(None, 10)	1010
drp2 (Dropout)	(None, 10)	0
fc2 (Dense)	(None, 1)	11
Total params: 31,921		
Trainable params: 31,921		
Non-trainable params: 0		

Train on 36868 samples, validate on 9218 samples

Epoch 1/30

36868/36868 [=====] - 15s 401us/step - loss: 0.0431 - val_loss: 0.0195

Epoch 2/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0238 - val_loss: 0.0191

Epoch 3/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0195 - val_loss: 0.0191

Epoch 4/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 5/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 6/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 7/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 8/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 9/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 10/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 11/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 12/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 13/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 14/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 15/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 16/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 17/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 18/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 19/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 20/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 21/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 22/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 23/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 24/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 25/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 26/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 27/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 28/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 29/30

36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

Epoch 30/30

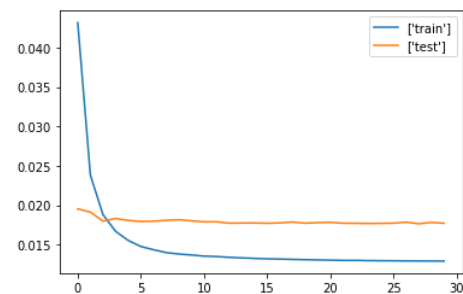
36868/36868 [=====] - 14s 392us/step - loss: 0.0191 - val_loss: 0.0191

pred=reg.predict(x_train)

err=np.mean(np.abs(y_train-pred))

print("MAE error for standard averaging:",err)

MAE error for standard averaging: 0.013842379968173548



pred=reg.predict(x_train)

err=np.mean(np.abs(y_train-pred))

print("MAE error for standard averaging:",err)

MAE error for standard averaging: 0.013842379968173548

All the above plots are based on the validation size of 0.20.