PS20

January 26, 2024

1 Mini-project #2 (RNN exercise)

1.1 Task: Weather prediction

1.2 Loading Jena climate dataset

[1]:
```
import pandas as pd
```

[2]:
```
data = pd.read_csv('jena_climate_2009_2016.csv')
data.describe()
```

[2]:

	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	\
count	420551.000000	420551.000000	420551.000000	420551.000000	
mean	989.212776	9.450147	283.492743	4.955854	
std	8.358481	8.423365	8.504471	6.730674	
min	913.600000	-23.010000	250.600000	-25.010000	
25%	984.200000	3.360000	277.430000	0.240000	
50%	989.580000	9.420000	283.470000	5.220000	
75%	994.720000	15.470000	289.530000	10.070000	
max	1015.350000	37.280000	311.340000	23.110000	

	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	\
count	420551.000000	420551.000000	420551.000000	420551.000000	
mean	76.008259	13.576251	9.533756	4.042412	
std	16.476175	7.739020	4.184164	4.896851	
min	12.950000	0.950000	0.790000	0.000000	
25%	65.210000	7.780000	6.210000	0.870000	
50%	79.300000	11.820000	8.860000	2.190000	
75%	89.400000	17.600000	12.350000	5.300000	
max	100.000000	63.770000	28.320000	46.010000	

	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	\
count	420551.000000	420551.000000	420551.000000	420551.000000	
mean	6.022408	9.640223	1216.062748	1.702224	
std	2.656139	4.235395	39.975208	65.446714	
min	0.500000	0.800000	1059.450000	-9999.000000	
25%	3.920000	6.290000	1187.490000	0.990000	
50%	5.590000	8.960000	1213.790000	1.760000	

```
75%          7.800000        12.490000    1242.770000       2.860000
max         18.130000        28.820000    1393.540000      28.490000

       max. wv (m/s)       wd (deg)
count  420551.000000  420551.000000
mean        3.056555     174.743738
std        69.016932      86.681693
min     -9999.000000       0.000000
25%         1.760000     124.900000
50%         2.960000     198.100000
75%         4.740000     234.100000
max        23.500000     360.000000
```

1.3 Data preprocessing

Wind speed (and maximum wind speed) is set to `-9999.00` for missing entries. Let us fill up the missing entries with the mean.

[3]:
```
wv = data['wv (m/s)']
wv_missing_idx = (wv == -9999.00)
wv_mean = wv[~wv_missing_idx].mean()
wv[wv_missing_idx] = wv_mean
```

```
C:\Users\chsuh\AppData\Local\Temp\ipykernel_18376\832945953.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  wv[wv_missing_idx] = wv_mean
```

[4]:
```
max_wv = data['max. wv (m/s)']
missing_idx = (max_wv == -9999.00)
max_wv_mean = max_wv[~missing_idx].mean()
max_wv[missing_idx] = max_wv_mean
```

```
C:\Users\chsuh\AppData\Local\Temp\ipykernel_18376\1423667913.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  max_wv[missing_idx] = max_wv_mean
```

1.4 Remove `date_time` column

[5]:
```
data.pop('Date Time')
data
```

[5]:
```
        p (mbar)  T (degC)  Tpot (K)  Tdew (degC)  rh (%)  VPmax (mbar)  \
0         996.52     -8.02    265.40        -8.90   93.30          3.33
1         996.57     -8.41    265.01        -9.28   93.40          3.23
2         996.53     -8.51    264.91        -9.31   93.90          3.21
3         996.51     -8.31    265.12        -9.07   94.20          3.26
4         996.51     -8.27    265.15        -9.04   94.10          3.27
…            …         …         …            …       …             …
420546   1000.07     -4.05    269.10        -8.13   73.10          4.52
420547    999.93     -3.35    269.81        -8.06   69.71          4.77
420548    999.82     -3.16    270.01        -8.21   67.91          4.84
420549    999.81     -4.23    268.94        -8.53   71.80          4.46
420550    999.82     -4.82    268.36        -8.42   75.70          4.27

        VPact (mbar)  VPdef (mbar)  sh (g/kg)  H2OC (mmol/mol)  rho (g/m**3)  \
0               3.11          0.22       1.94             3.12       1307.75
1               3.02          0.21       1.89             3.03       1309.80
2               3.01          0.20       1.88             3.02       1310.24
3               3.07          0.19       1.92             3.08       1309.19
4               3.08          0.19       1.92             3.09       1309.00
…                  …             …          …                …             …
420546          3.30          1.22       2.06             3.30       1292.98
420547          3.32          1.44       2.07             3.32       1289.44
420548          3.28          1.55       2.05             3.28       1288.39
420549          3.20          1.26       1.99             3.20       1293.56
420550          3.23          1.04       2.01             3.23       1296.38

        wv (m/s)  max. wv (m/s)  wd (deg)
0           1.03           1.75     152.3
1           0.72           1.50     136.1
2           0.19           0.63     171.6
3           0.34           0.50     198.0
4           0.32           0.63     214.3
…              …              …         …
420546      0.67           1.52     240.0
420547      1.14           1.92     234.3
420548      1.08           2.00     215.2
420549      1.49           2.16     225.8
420550      1.23           1.96     184.9

[420551 rows x 14 columns]
```

1.5 Downsampling

We sample every 60 minutes to contruct a dataset for training.

[6]:
```
data = data[0::6]
#data = data[::6]
data
```

[6]:

	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	\
0	996.52	-8.02	265.40	-8.90	93.30	3.33	
6	996.50	-7.62	265.81	-8.30	94.80	3.44	
12	996.63	-8.85	264.57	-9.70	93.50	3.12	
18	996.87	-8.84	264.56	-9.69	93.50	3.13	
24	997.05	-9.23	264.15	-10.25	92.20	3.03	
…	…	…	…	…	…	…	
420522	1002.08	-1.40	271.59	-6.10	70.20	5.51	
420528	1001.42	-2.15	270.90	-7.08	68.77	5.21	
420534	1001.05	-2.61	270.47	-6.97	71.80	5.04	
420540	1000.51	-3.22	269.90	-7.63	71.40	4.81	
420546	1000.07	-4.05	269.10	-8.13	73.10	4.52	

	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	\
0	3.11	0.22	1.94	3.12	1307.75	
6	3.26	0.18	2.04	3.27	1305.68	
12	2.92	0.20	1.82	2.93	1312.11	
18	2.92	0.20	1.83	2.93	1312.37	
24	2.79	0.24	1.74	2.80	1314.62	
…	…	…	…	…	…	
420522	3.87	1.64	2.40	3.86	1282.68	
420528	3.59	1.63	2.23	3.58	1285.50	
420534	3.62	1.42	2.25	3.61	1287.20	
420540	3.44	1.38	2.14	3.44	1289.50	
420546	3.30	1.22	2.06	3.30	1292.98	

	wv (m/s)	max. wv (m/s)	wd (deg)
0	1.03	1.75	152.3
6	0.18	0.63	166.5
12	0.16	0.50	158.3
18	0.07	0.25	129.3
24	0.10	0.38	203.9
…	…	…	…
420522	1.08	1.68	207.5
420528	0.79	1.24	184.3
420534	0.77	1.64	129.1
420540	0.85	1.54	207.8
420546	0.67	1.52	240.0

[70092 rows x 14 columns]

1.6 Take temperature in celsius as label while taking everything as features

[7]:
```
features = data
labels = data[['T (degC)']]
#labels = data['T (degC)']
```

[8]:
```
print(features.shape)
print(labels.shape)
print(type(features))
print(type(labels))
```

```
(70092, 14)
(70092, 1)
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
```

1.7 Normalization

[9]:
```
from sklearn.preprocessing import StandardScaler

std_scaler = StandardScaler()
features = std_scaler.fit_transform(features)
```

[10]:
```
print(type(features))
print(features)
print(features.shape)
```

```
<class 'numpy.ndarray'>
[[ 0.87420457 -2.07391772 -2.12735513 … -0.71190538 -0.76237653
  -0.2618485 ]
 [ 0.87181184 -2.02643323 -2.07914744 … -1.26284569 -1.24217323
  -0.09825609]
 [ 0.8873646  -2.17244806 -2.2249463  … -1.27580899 -1.29786392
  -0.19272494]
 …
 [ 1.41615816 -1.43168989 -1.53122591 … -0.8804283  -0.80949942
  -0.52912624]
 [ 1.35155442 -1.50410375 -1.59824636 … -0.82857509 -0.85233841
   0.37754438]
 [ 1.29891433 -1.60263408 -1.69231014 … -0.9452448  -0.86090621
   0.74850745]]
(70092, 14)
```

1.8 Data split (train:val:test=7:2:1)

Let us split chronologically, not randomly.

[11]:
```
from sklearn.model_selection import train_test_split
```

```
## For chronological splitting, we set "shuffle=False"
X_rest, X_test, y_rest, y_test = train_test_split(features,
                                                  labels,
                                                  test_size=0.1,
                                                  shuffle=False)
X_train, X_val, y_train, y_val = train_test_split(X_rest,
                                                  y_rest,
                                                  test_size=2/9,
                                                  shuffle=False)
```

[12]:
```
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)
```

```
(49063, 14)
(14019, 14)
(7010, 14)
```

1.9 Set T_{window} and batch size

[13]:
```
T = 24
batch_size = 16
```

1.10 Construct $\{(x_T^{(i)}, y_T^{(i)})\}_{i=1}^{m_T}$

[14]:
```
from tensorflow.keras.preprocessing import timeseries_dataset_from_array

# Train batch dataset
dataset_train = timeseries_dataset_from_array(X_train[:-T],
                                              y_train[T:],
                                              sequence_length = T,
                                              sequence_stride = 1,
                                              batch_size = batch_size,
                                              shuffle = True)
# validation batch dataset
dataset_val = timeseries_dataset_from_array(X_val[:-T],
                                            y_val[T:],
                                            sequence_length = T,
                                            sequence_stride = 1,
                                            batch_size=batch_size,
                                            shuffle = False)
# test batch dataset
dataset_test = timeseries_dataset_from_array(X_test[:-T],
                                             y_test[T:],
                                             sequence_length = T,
                                             sequence_stride = 1,
                                             batch_size=batch_size,
```

```
                                                shuffle = False)
```

[15]:
```
print(type(dataset_train))
```

```
<class 'tensorflow.python.data.ops.dataset_ops.BatchDataset'>
```

[16]:
```
print(dataset_train.take)
```

```
<bound method DatasetV2.take of <BatchDataset
element_spec=(TensorSpec(shape=(None, None, 14), dtype=tf.float64, name=None),
TensorSpec(shape=(None, 1), dtype=tf.float64, name=None))>>
```

[17]:
```
print(len(dataset_train))
print(len(X_train[:-T])//batch_size)
print(len(dataset_val))
print(len(X_val[:-T])//batch_size)
print(len(dataset_test))
print(len(X_test[:-T])//batch_size)
```

```
3064
3064
874
874
436
436
```

[18]:
```
print(dataset_train.take)
print(len(dataset_train))
print(len(dataset_val))
print(len(dataset_test))
```

```
<bound method DatasetV2.take of <BatchDataset
element_spec=(TensorSpec(shape=(None, None, 14), dtype=tf.float64, name=None),
TensorSpec(shape=(None, 1), dtype=tf.float64, name=None))>>
3064
874
436
```

[19]:
```
for batch in dataset_train.take(5):
    inputs, labels =  batch
    print(inputs.shape)
    print(labels.shape)
```

```
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
(16, 24, 14)
```

```
(16, 1)
(16, 24, 14)
(16, 1)
```

[20]:
```
print(type(inputs))
print(inputs)
```

```
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(
[[[-0.45256479 -0.261197   -0.22373935 … -1.21099248 -1.28929612
   -1.60226942]
  [-0.4035138  -0.34785621 -0.31427574 … -1.20451083 -1.30643172
   -1.00607877]
  [-0.37121193 -0.10568527 -0.07558889 … -1.01654296 -1.01512658
   -1.42277859]
  …
  [-0.18816801  0.45107047  0.46292626 … -0.89987325 -0.98085539
   -1.87507705]
  [-0.17381162  0.4154571   0.42647654 … -0.69894208 -0.86090621
    2.10217706]
  [-0.1295461   0.2825005   0.29008406 … -0.24522653 -0.50105869
   -1.74523997]]

 [[ 0.0857997  -0.12824041 -0.13437876 … -0.06374031  0.08155159
   -0.10862462]
  [ 0.07503241  0.0154002   0.00906851 …  1.31685129  1.16537806
    0.35219907]
  [ 0.03555234  0.02964555  0.02552967 …  3.55302078  3.34159881
    0.40288968]
  …
  [-0.42265565  0.01183886  0.04551823 … -0.26467148 -0.48392309
    0.18860666]
  [-0.42744111  0.01183886  0.04551823 …  0.00755785 -0.10693997
    0.21510403]
  [-0.42624475  0.31930099  0.35122552 …  1.61500722  1.10540348
    0.31648524]]

 [[-0.25037902 -0.18997026 -0.16965267 …  0.31219543  0.86550513
    0.19667108]
  [-0.20970259 -0.19471871 -0.17670746 …  1.26499808  1.26819164
    0.30035641]
  [-0.17261526 -0.17809913 -0.16377369 …  3.50764922  3.56436156
    0.35680731]
  …
  [ 0.3633565  -0.36922424 -0.39658155 …  1.16777332  1.04542889
    0.22547256]
  [ 0.32387643 -0.31580418 -0.34014328 …  1.21962653  1.07113228
    0.3372223 ]
```

```
 [ 0.2939673  -0.34192065 -0.36365922 …  0.38997524  0.4970898
   0.2807714 ]]

…

[[-0.11758244  0.70273831  0.70749209 … -0.9452448  -1.03226218
   0.29690023]
 [-0.10322606  0.6730605   0.67692136 … -1.12673102 -1.27216052
   0.79113364]
 [-0.08767331  0.68018317  0.68280035 … -0.8804283  -1.06653337
   0.23699315]
 …
 [ 0.19466896  0.70748676  0.68632774 … -0.71838703 -0.73238924
   1.0042646 ]
 [ 0.2520945   0.62082755  0.59579135 … -1.11376772 -1.11794016
  -1.77208295]
 [ 0.21620354  0.58996263  0.56757222 … -1.02302461 -1.08366896
  -1.4423636 ]]

[[-0.05537144  1.39838621  1.3929819  …  0.68164952  0.80124664
   0.03999102]
 [-0.08408421  1.34496615  1.34242262 …  0.92795225  0.64702627
   0.15750106]
 [-0.0948515   1.20607399  1.20603013 …  0.16959912 -0.14121116
   0.37063202]
 …
 [-0.6846597   1.53371703  1.58110946 …  0.14367251  0.62989068
  -1.46920658]
 [-0.76481619  1.63343448  1.68693122 … -0.03781371 -0.17548236
   0.45358029]
 [-0.79472533  1.67973186  1.7363147  … -0.05725866 -0.02126199
  -0.8920249 ]]

[[-0.42624475 -1.37945694 -1.33839516 …  0.24737892 -0.01269419
  -0.08327932]
 [-0.48367029 -1.41625742 -1.37014169 … -0.24522653 -0.38539341
   0.22547256]
 [-0.55425586 -1.39014095 -1.33957096 … -0.34893294 -0.59958837
   0.49044618]
 …
 [-1.5520247  -0.54016843 -0.41304271 … -0.64708887 -0.81378332
   0.9109478 ]
 [-1.61303935 -0.55797512 -0.42597648 … -1.04895121 -0.97657149
  -0.23765525]
 [-1.66089397 -0.61851785 -0.48241475 … -1.10728607 -1.08366896
  -1.38326296]]], shape=(16, 24, 14), dtype=float64)
```

[21]:
```
print(type(labels))
print(labels)
```

```
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(
[[ 10.34]
 [ 12.51]
 [  6.46]
 [ -4.75]
 [  6.48]
 [  0.77]
 [ 11.85]
 [ 27.94]
 [  6.15]
 [ -1.23]
 [-15.31]
 [ 10.27]
 [ -0.91]
 [ 14.12]
 [ 22.16]
 [  4.54]], shape=(16, 1), dtype=float64)
```

[ ]:

[ ]: