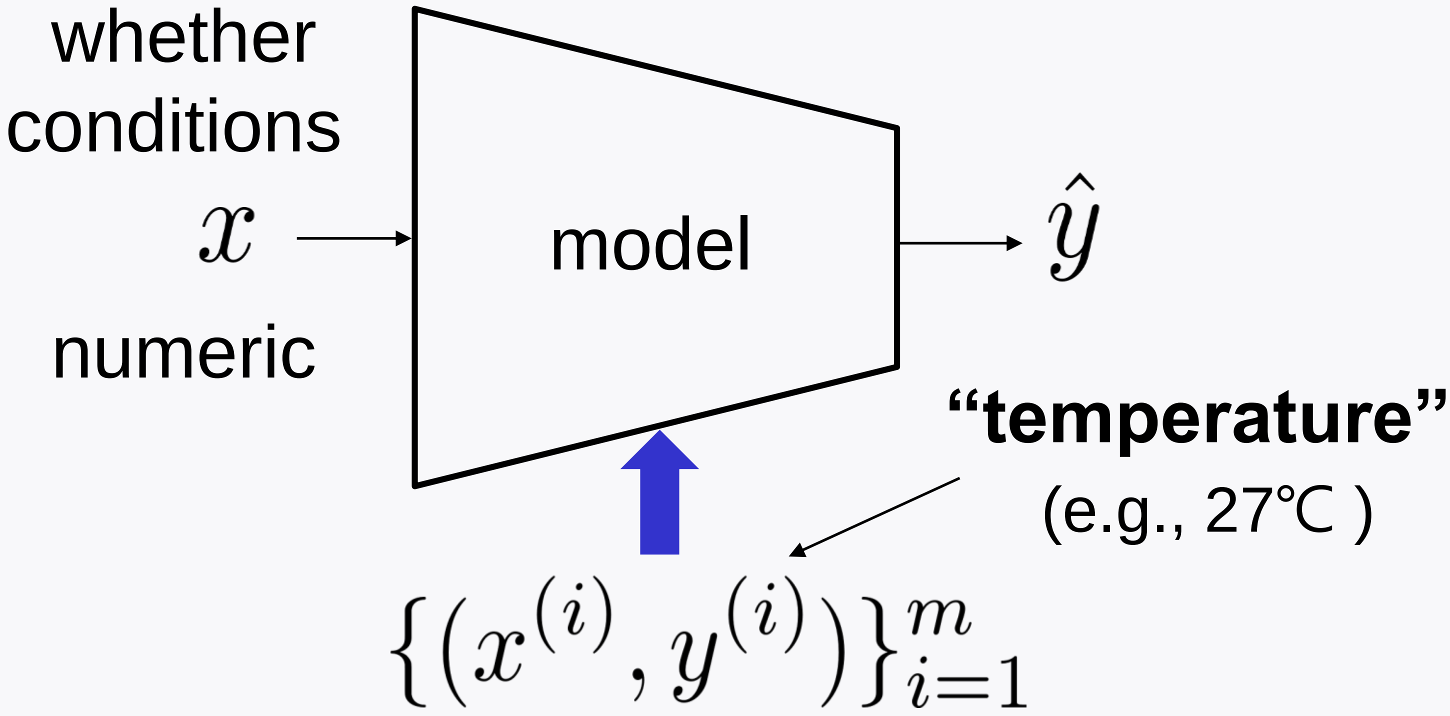
Mini-project #2

Practice Session 21

Changho Suh

January 30, 2024

Recap: Weather prediction



whether conditions

x

numeric

model

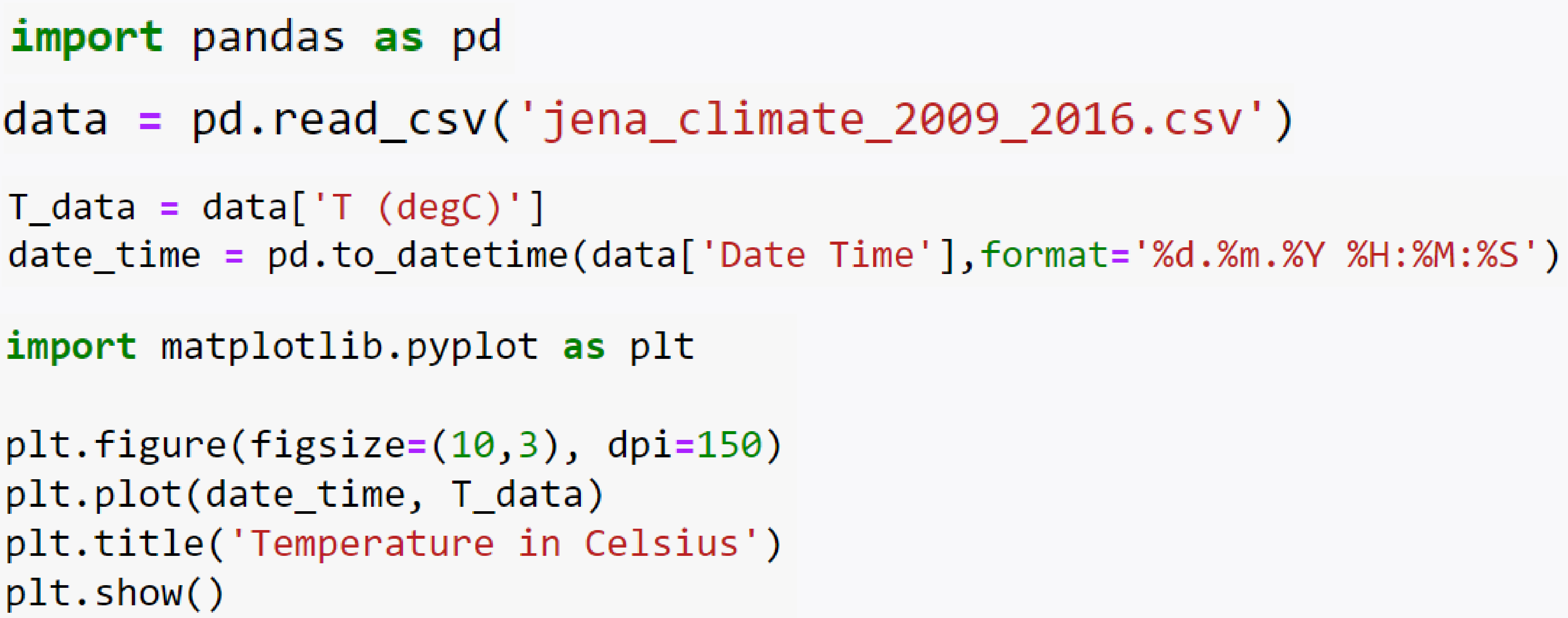
$\hat{y}$

“temperature”

(e.g., 27℃)

$$\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$$

Recap: Data load & visualization



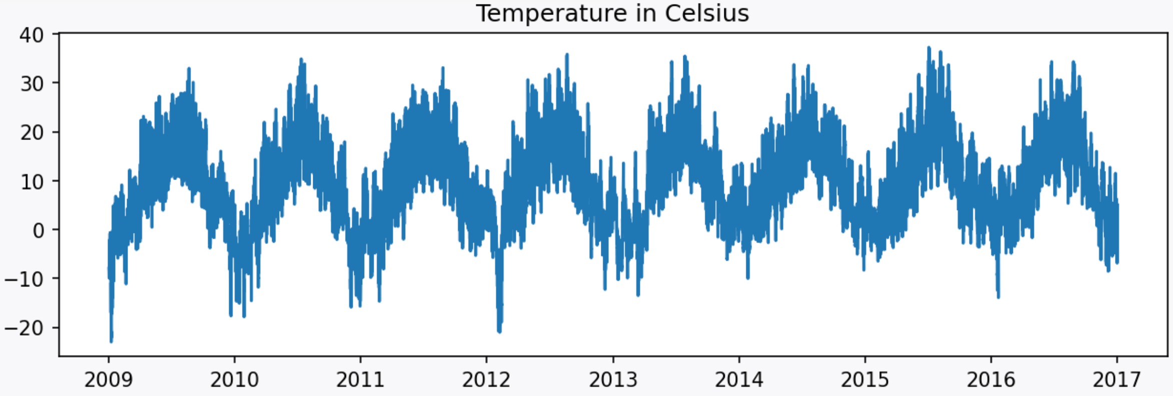
```
import pandas as pd
```

```
data = pd.read_csv('jena_climate_2009_2016.csv')
```

```
T_data = data['T (degC)']
date_time = pd.to_datetime(data['Date Time'],format='%d.%m.%Y %H:%M:%S')
```

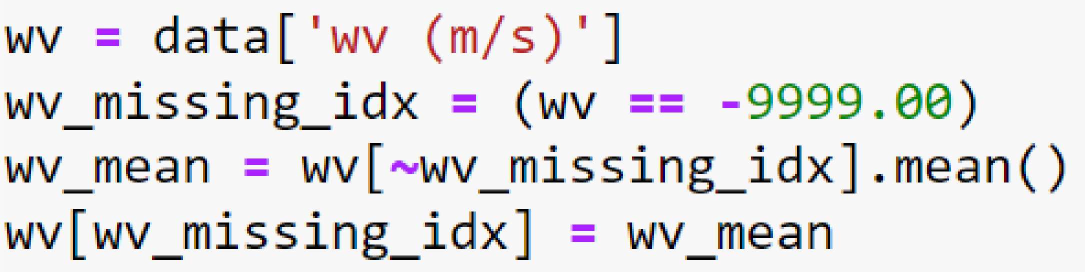
```
import matplotlib.pyplot as plt

plt.figure(figsize=(10,3), dpi=150)
plt.plot(date_time, T_data)
plt.title('Temperature in Celsius')
plt.show()
```

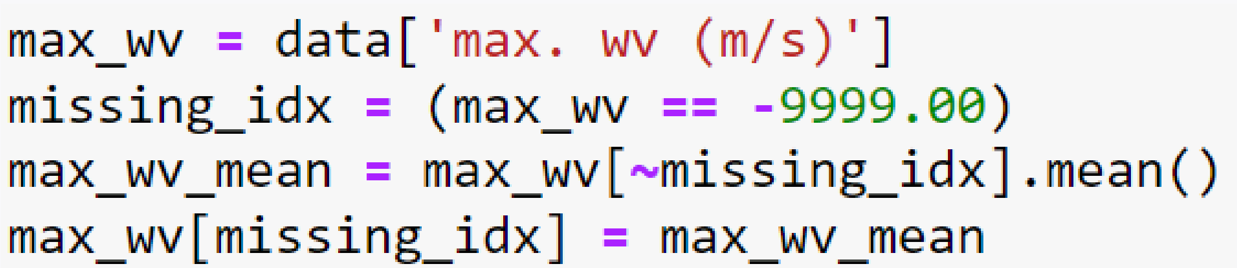


Recap: Preprocessing

# fill up missing entries in wind speed

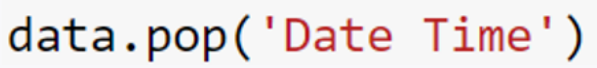


```
wv = data['wv (m/s)']
wv_missing_idx = (wv == -9999.00)
wv_mean = wv[~wv_missing_idx].mean()
wv[wv_missing_idx] = wv_mean
```



```
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
```

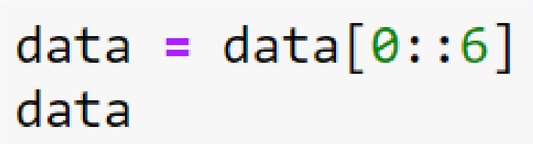
# remove ‘data_time’ column



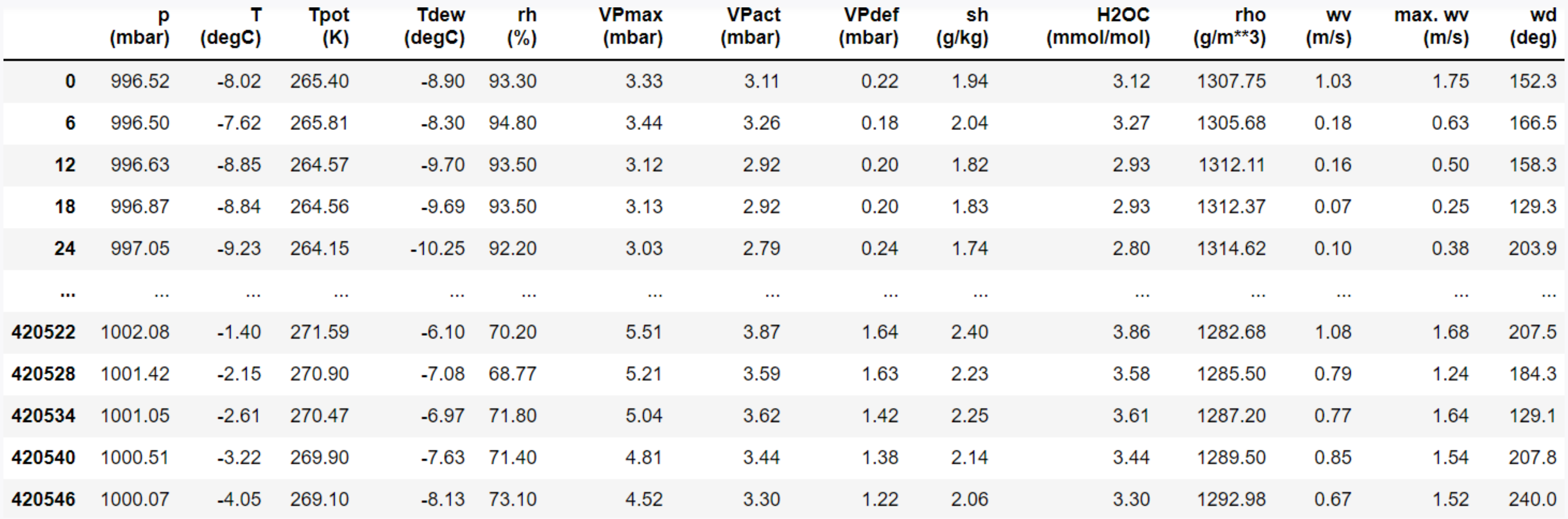
```
data.pop('Date Time')
```

Downsample

Sample every 60 minutes (instead of 10 minutes).



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



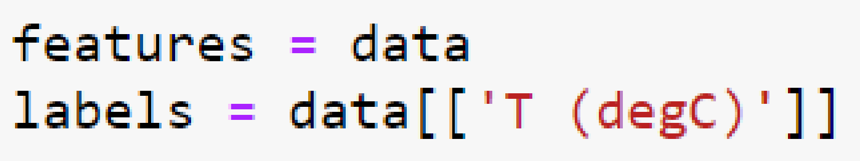
	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	max. wv (m/s)	wd (deg)
0	996.52	-8.02	265.40	-8.90	93.30	3.33	3.11	0.22	1.94	3.12	1307.75	1.03	1.75	152.3
6	996.50	-7.62	265.81	-8.30	94.80	3.44	3.26	0.18	2.04	3.27	1305.68	0.18	0.63	166.5
12	996.63	-8.85	264.57	-9.70	93.50	3.12	2.92	0.20	1.82	2.93	1312.11	0.16	0.50	158.3
18	996.87	-8.84	264.56	-9.69	93.50	3.13	2.92	0.20	1.83	2.93	1312.37	0.07	0.25	129.3
24	997.05	-9.23	264.15	-10.25	92.20	3.03	2.79	0.24	1.74	2.80	1314.62	0.10	0.38	203.9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
420522	1002.08	-1.40	271.59	-6.10	70.20	5.51	3.87	1.64	2.40	3.86	1282.68	1.08	1.68	207.5
420528	1001.42	-2.15	270.90	-7.08	68.77	5.21	3.59	1.63	2.23	3.58	1285.50	0.79	1.24	184.3
420534	1001.05	-2.61	270.47	-6.97	71.80	5.04	3.62	1.42	2.25	3.61	1287.20	0.77	1.64	129.1
420540	1000.51	-3.22	269.90	-7.63	71.40	4.81	3.44	1.38	2.14	3.44	1289.50	0.85	1.54	207.8
420546	1000.07	-4.05	269.10	-8.13	73.10	4.52	3.30	1.22	2.06	3.30	1292.98	0.67	1.52	240.0

$$m = 70,092$$

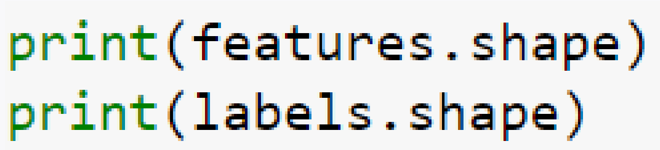
Features and label

Label: Temperature in Celsius

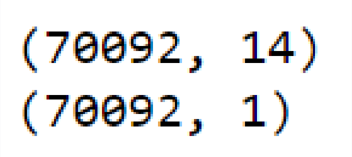
Features: Everything



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

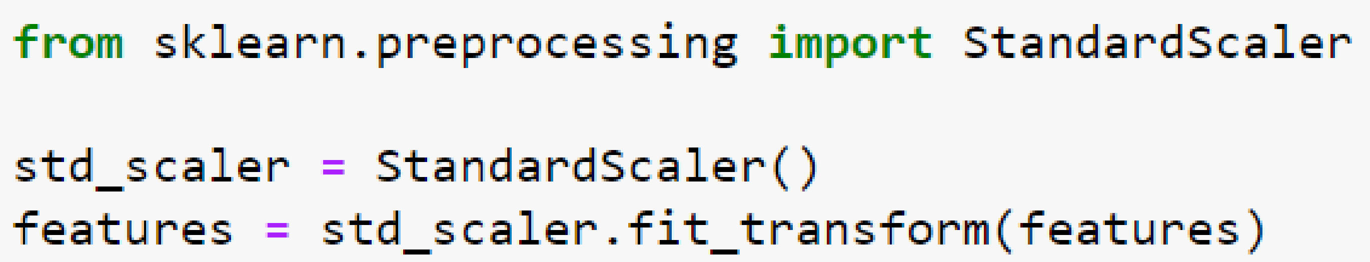


```
print(features.shape)
print(labels.shape)
```



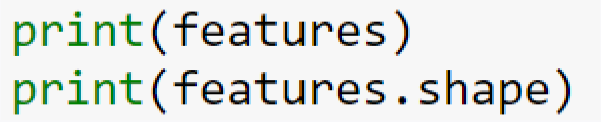
```
(70092, 14)
(70092, 1)
```

Normalization

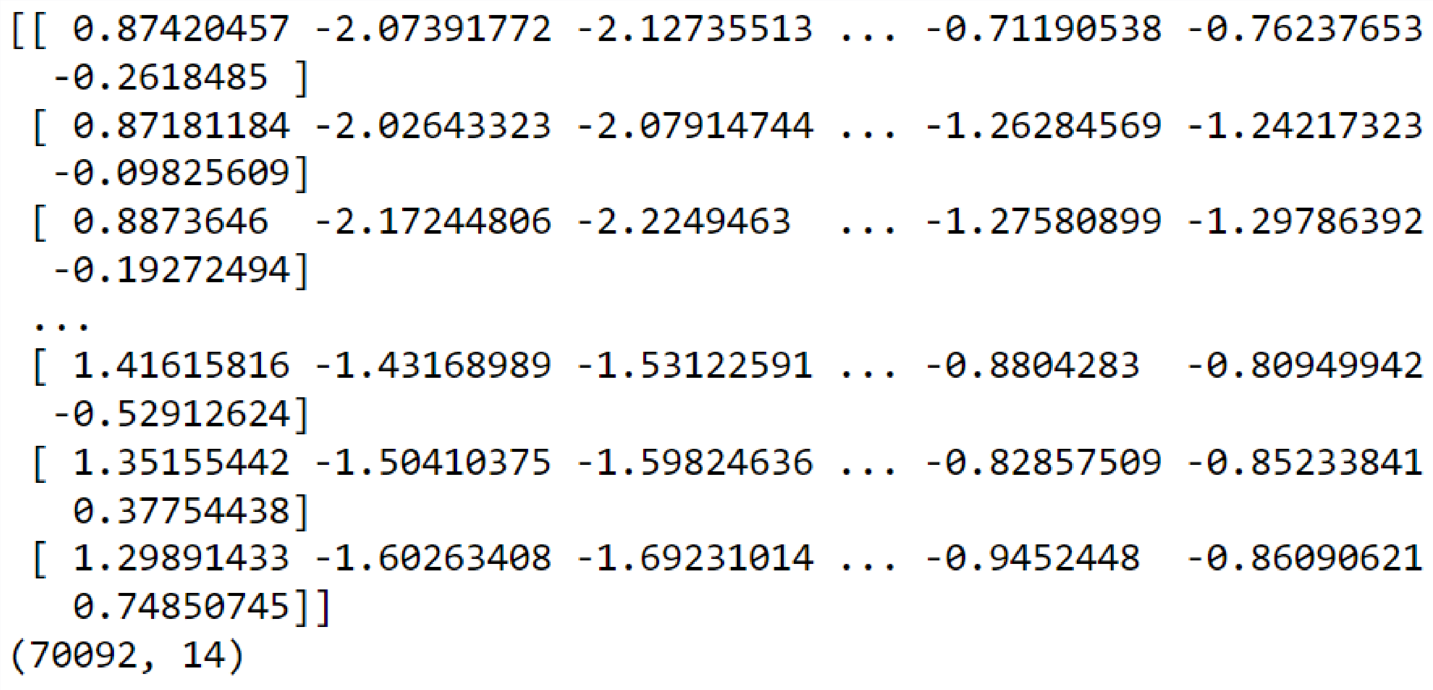


```
from sklearn.preprocessing import StandardScaler

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



```
print(features)
print(features.shape)
```

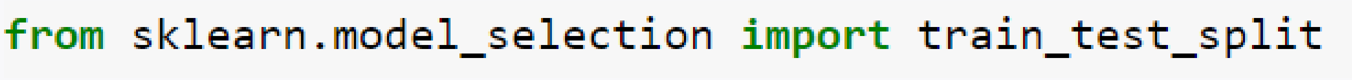


```
[[ 0.87420457 -2.07391772 -2.12735513 ... -0.71190538 -0.76237653
  -0.2618485 ]
 [ 0.87181184 -2.02643323 -2.07914744 ... -1.26284569 -1.24217323
  -0.09825609]
 [ 0.8873646  -2.17244806 -2.2249463  ... -1.27580899 -1.29786392
  -0.19272494]
 ...
 [ 1.41615816 -1.43168989 -1.53122591 ... -0.8804283  -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)
```

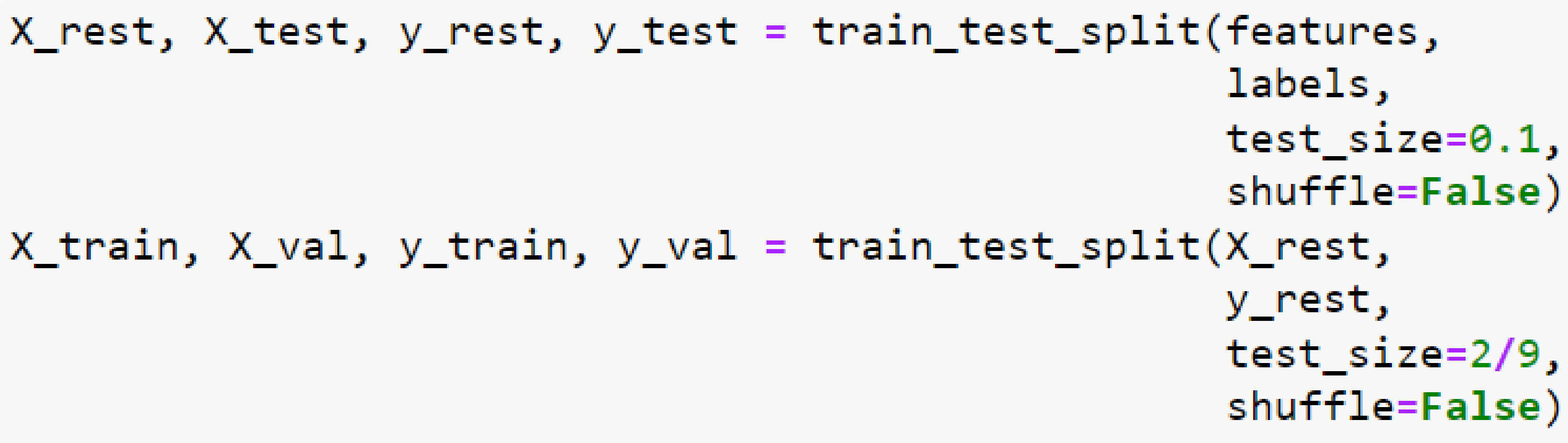
Data split

Split dataset into **train/val/test** sets with:

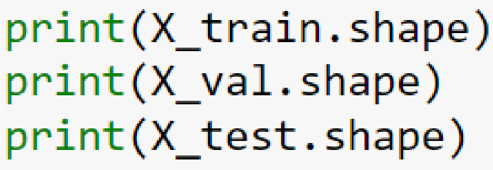
7:2:1 (in chronological order)



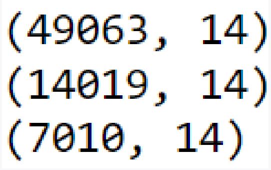
```
from sklearn.model_selection import train_test_split
```



```
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)
```



```
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)
```

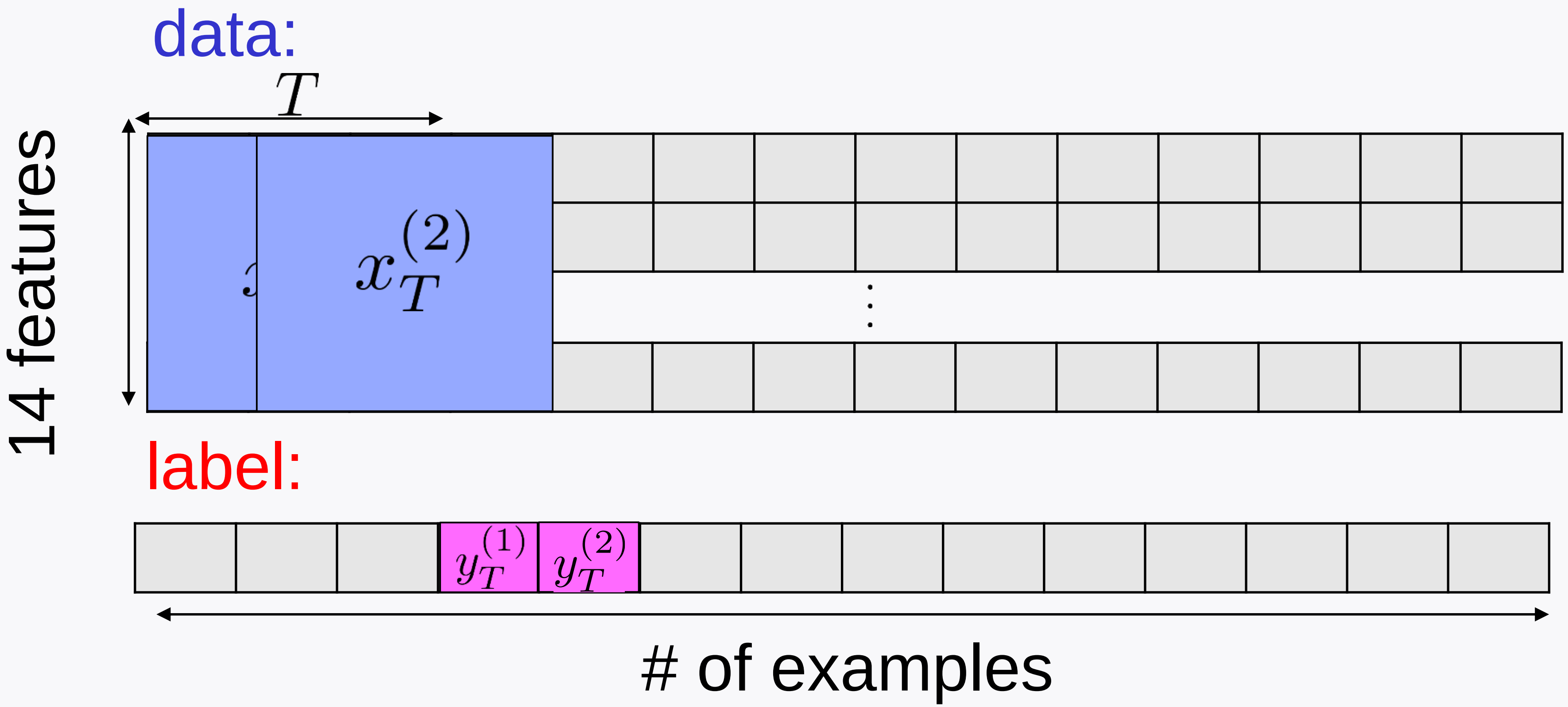


```
(49063, 14)
(14019, 14)
(7010, 14)
```

Time window *T*

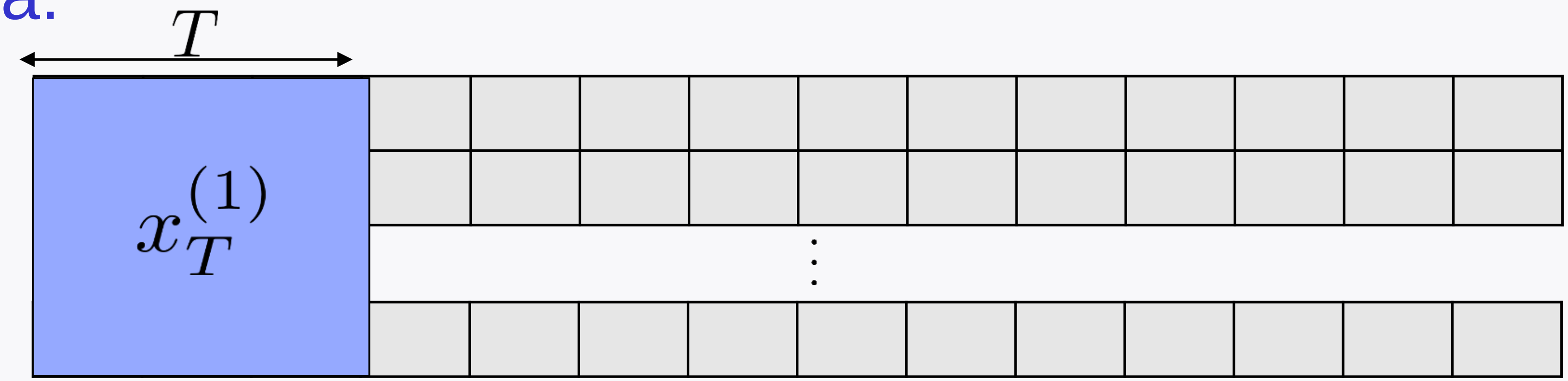
Generate time series dataset:

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

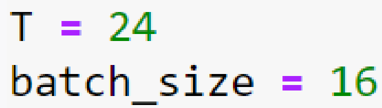


Code: Time series data generation

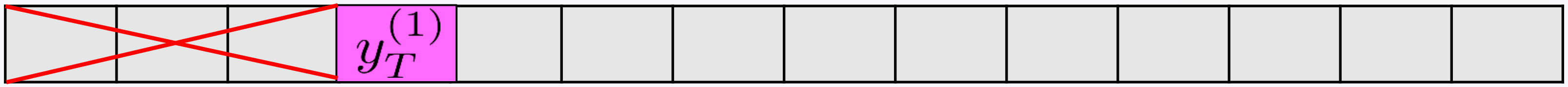
data:

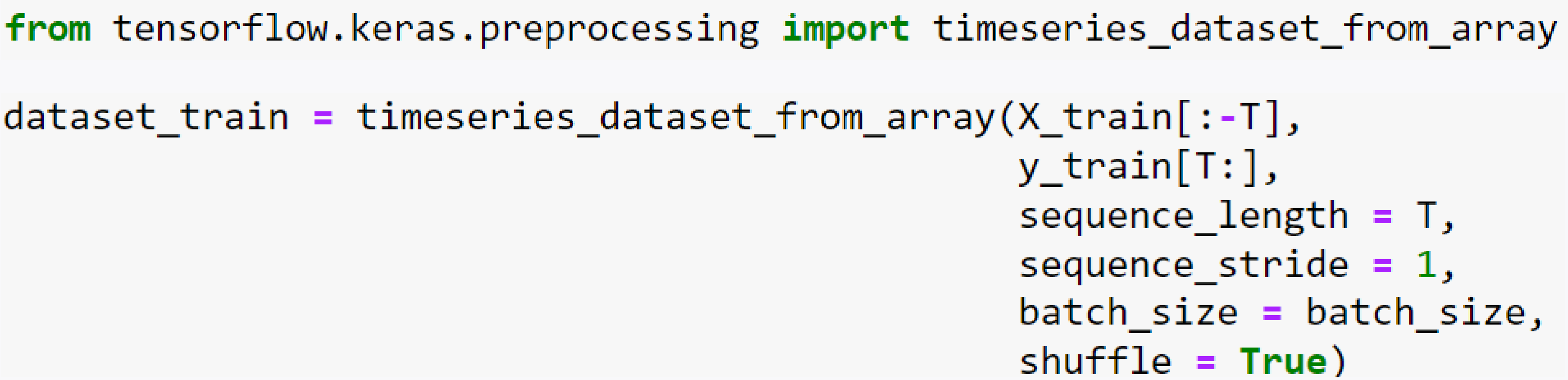


label:



```
T = 24
batch_size = 16
```

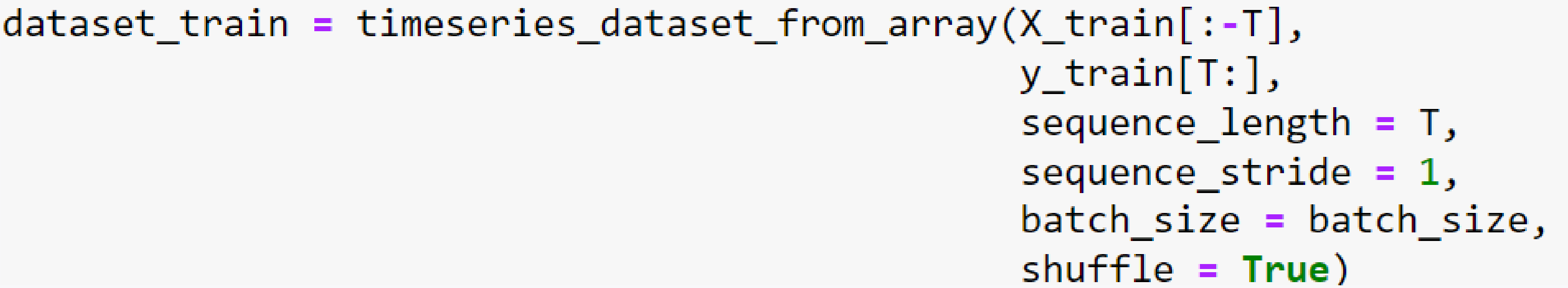




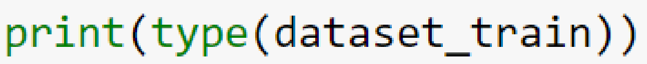
```
from tensorflow.keras.preprocessing import timeseries_dataset_from_array

dataset_train = timeseries_dataset_from_array(X_train[:-T],
                                              y_train[T:],
                                              sequence_length = T,
                                              sequence_stride = 1,
                                              batch_size = batch_size,
                                              shuffle = True)
```

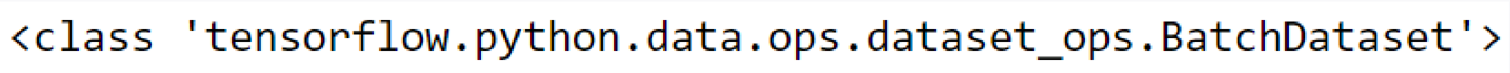
dataset_train



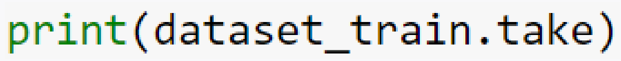
```
dataset_train = timeseries_dataset_from_array(X_train[:-T],
                                              y_train[T:],
                                              sequence_length = T,
                                              sequence_stride = 1,
                                              batch_size = batch_size,
                                              shuffle = True)
```



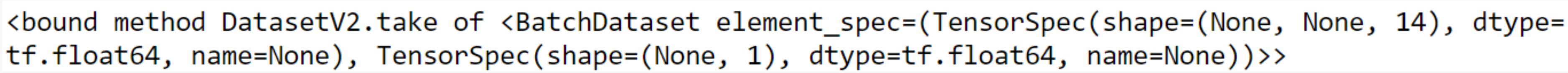
```
print(type(dataset_train))
```



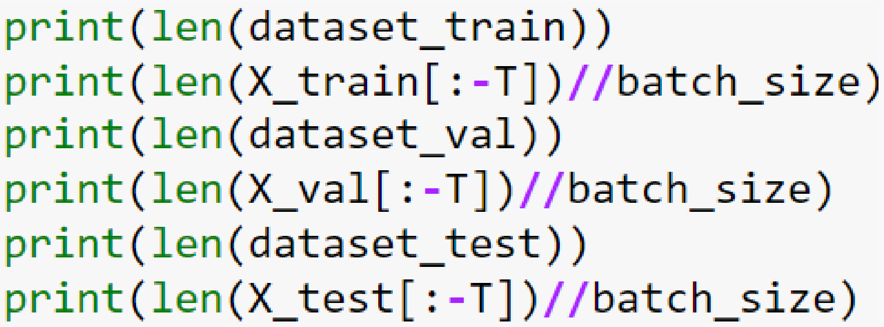
```
<class 'tensorflow.python.data.ops.dataset_ops.BatchDataset'>
```



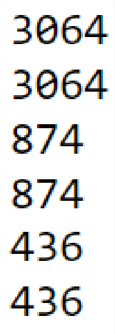
```
print(dataset_train.take)
```



```
<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))>>
```

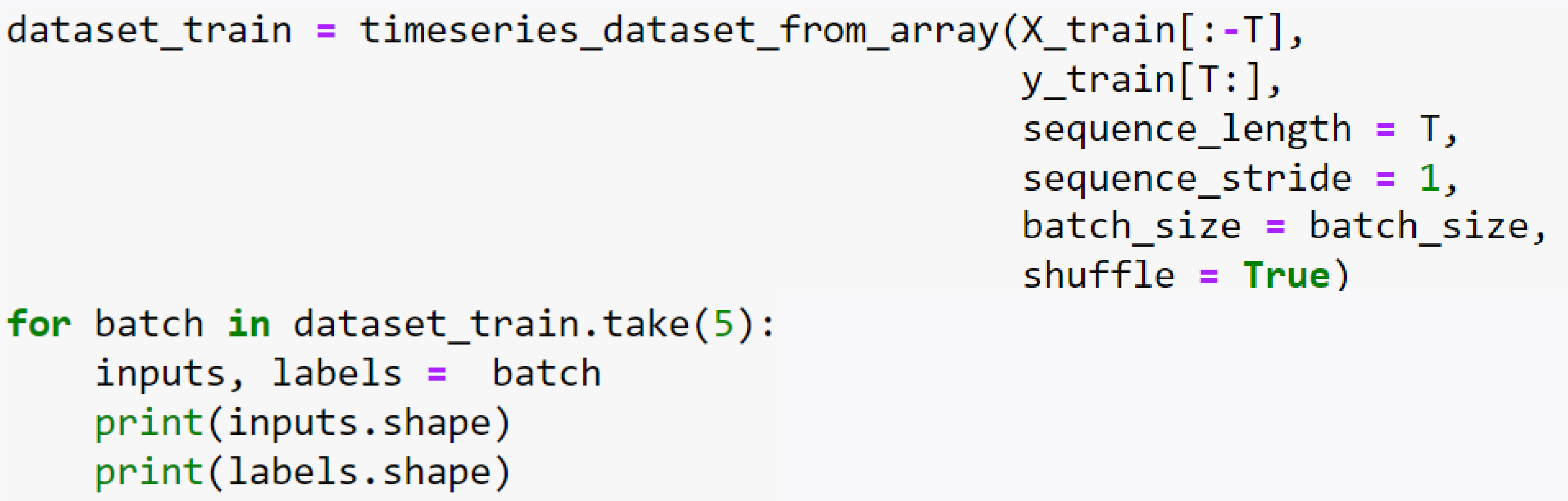


```
print(len(dataset_train))
print(len(X_train[:-T])//batch_size)
print(len(dataset_val))
print(len(X_val[:-T])//batch_size)
print(len(dataset_test))
print(len(X_test[:-T])//batch_size)
```

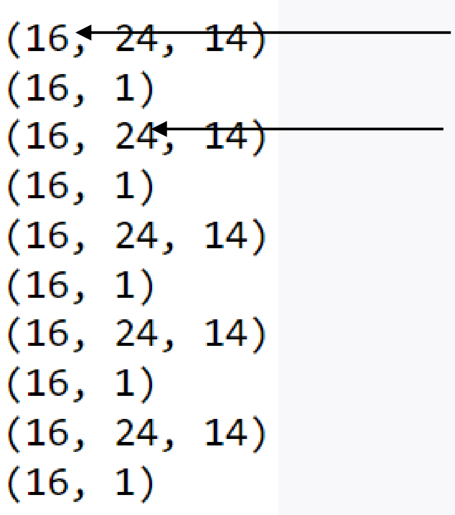


```
3064
3064
874
874
436
436
```

dataset_train



```
dataset_train = timeseries_dataset_from_array(X_train[:-T],
                                              y_train[T:],
                                              sequence_length = T,
                                              sequence_stride = 1,
                                              batch_size = batch_size,
                                              shuffle = True)
for batch in dataset_train.take(5):
    inputs, labels =  batch
    print(inputs.shape)
    print(labels.shape)
```

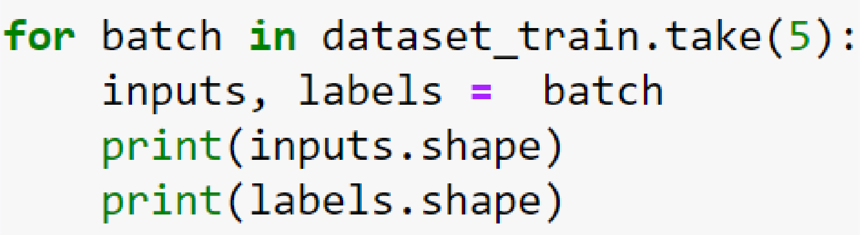


```
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
(16, 24, 14)
(16, 1)
```

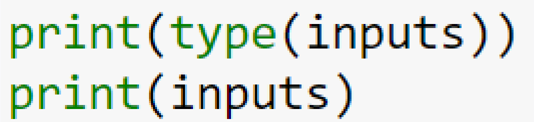
batch_size

T

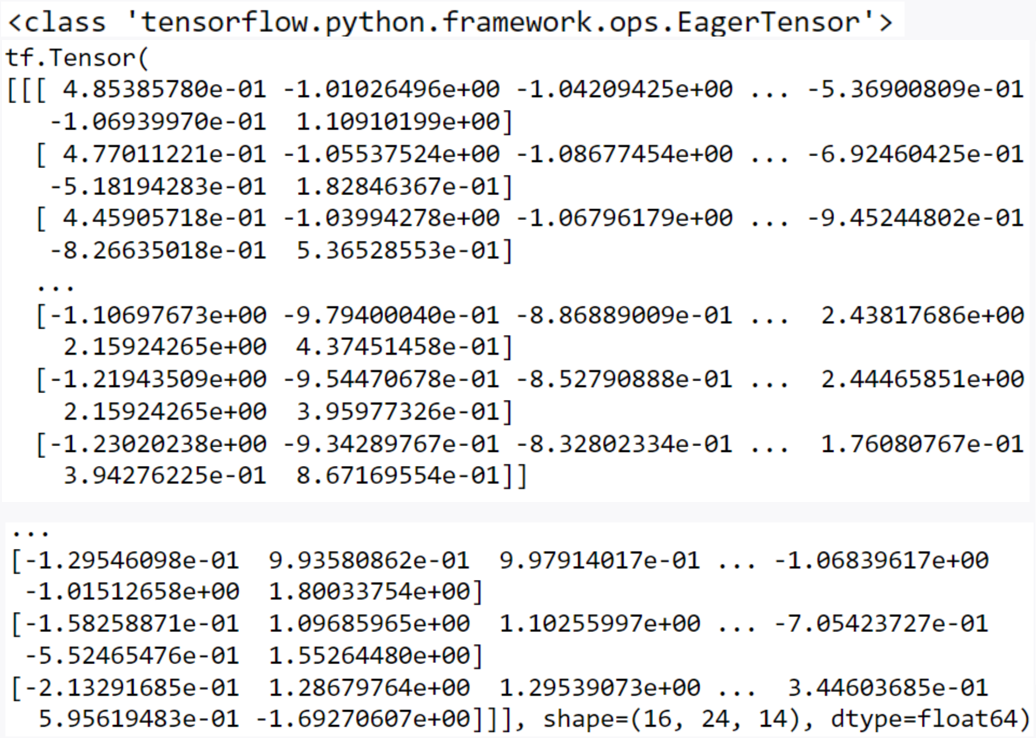
dataset_train



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

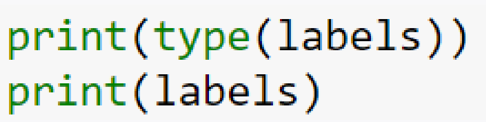


```
print(type(inputs))
print(inputs)
```

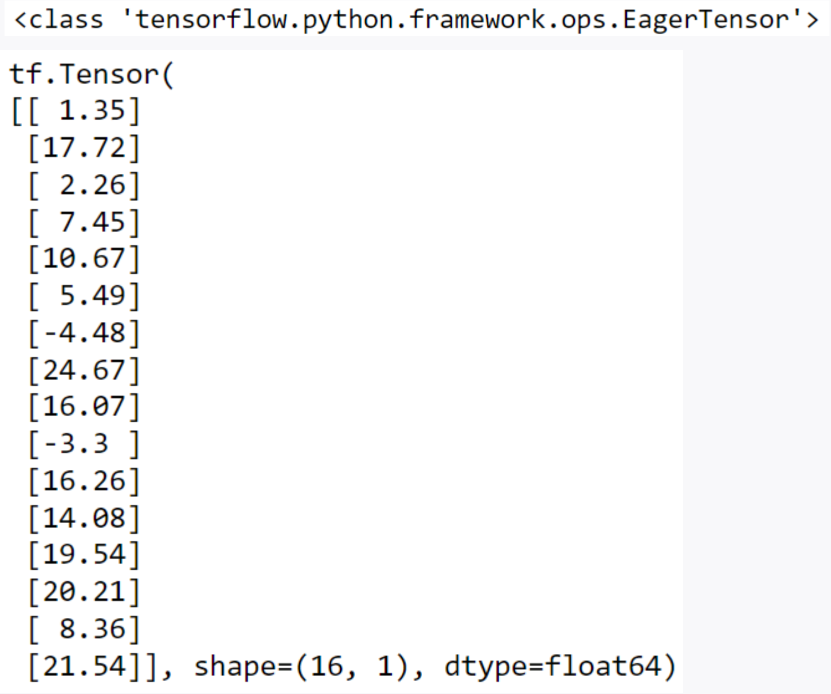


```
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(
[[[ 4.85385780e-01 -1.01026496e+00 -1.04209425e+00 ... -5.36900809e-01
   -1.06939970e-01  1.10910199e+00]
  [ 4.77011221e-01 -1.05537524e+00 -1.08677454e+00 ... -6.92460425e-01
   -5.18194283e-01  1.82846367e-01]
  [ 4.45905718e-01 -1.03994278e+00 -1.06796179e+00 ... -9.45244802e-01
   -8.26635018e-01  5.36528553e-01]
  ...
  [-1.10697673e+00 -9.79400040e-01 -8.86889009e-01 ...  2.43817686e+00
    2.15924265e+00  4.37451458e-01]
  [-1.21943509e+00 -9.54470678e-01 -8.52790888e-01 ...  2.44465851e+00
    2.15924265e+00  3.95977326e-01]
  [-1.23020238e+00 -9.34289767e-01 -8.32802334e-01 ...  1.76080767e-01
    3.94276225e-01  8.67169554e-01]]

 ...
 [-1.29546098e-01  9.93580862e-01  9.97914017e-01 ... -1.06839617e+00
  -1.01512658e+00  1.80033754e+00]
 [-1.58258871e-01  1.09685965e+00  1.10255997e+00 ... -7.05423727e-01
  -5.52465476e-01  1.55264480e+00]
 [-2.13291685e-01  1.28679764e+00  1.29539073e+00 ...  3.44603685e-01
   5.95619483e-01 -1.69270607e+00]]], shape=(16, 24, 14), dtype=float64)
```



```
print(type(labels))
print(labels)
```



```
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(
[[ 1.35]
 [17.72]
 [ 2.26]
 [ 7.45]
 [10.67]
 [ 5.49]
 [-4.48]
 [24.67]
 [16.07]
 [-3.3 ]
 [16.26]
 [14.08]
 [19.54]
 [20.21]
 [ 8.36]
 [21.54]], shape=(16, 1), dtype=float64)
```

Look ahead

Will train DNN and RNN models.