LAB Logbook

Lab 1

Lab Logbook Requirement:

1) Create a vector using np.arange.
Determine the number of the vector elements using the following method: Take the last two digits from your SID. It should be from 00 to 99. If this number is 10 or more, it becomes the required number of the vector elements. If it is less than 10, add 100 to your number.

For example, if your SID is 2287467, and the last two digits are 67, which is greater than 10. The required number is 67.
If your SID is 2287407, and the last two digits are 07, which is less than 10. The required number is 107.

Then,

2. *Change matrix a to 2-d array with 1 row. Print the array. You should have the two sets of brackets for a 2-d array with one row.*
3. *Save it in another array. Print the array.*
4. *Check the shape attribute value.*
5. *Add the code and result to your Lab Logbook*

NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER NOTEBOOK AND LAB LOGBOOK ON GITHUB OR ONEDRIVE.

1. Created a vector with np.arange. The last two digits of my SID is 31. So the range is from 1 to 32 in order to get 31 elements.

```
[17]: a = np.arange(1,32)
      print(a)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 25 26 27 28 29 30 31]
```

2. Changed the matrix a to 2D array with one row using reshape. The array has two sets of brackets.

[23]:
```
a = a.reshape(31,1)
print(a)
```

```
[[ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]
 [24]
 [25]
 [26]
 [27]
 [28]
 [29]
 [30]
 [31]]
```

3. Saved 2D array a into b. Then show contents of b.

```
[25]: b = a
      print(b)
```

```
[[ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]
 [24]
 [25]
 [26]
 [27]
 [28]
 [29]
 [30]
 [31]]
```

4. Check the shape attribute value for b.

```
[15]: b.shape
```

```
[15]: (31,)
```

Lab 2

Lab Logbook Requirement:

```
<html> <h3 style="font-style:italic; color:blue;">

1) Determine a number (n) equal to the last digit of your SID.
2) Group by "relationship" and "hours-per-week".
3) Reduce all "hours-per-week" column values in the original DataFrame by the value 'n'.
4) Group by "relationship" and reduced "hours-per-week".
5) Add the code and result to your Lab Logbook.
```

NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER
GITHUB OR ONEDRIVE.

[90]:
```
n = 31

Group_by_relationship_hpw = data.groupby(["relationship","hours-per-week"])
Group_by_relationship_hpw.size().unstack()
```

[90]:

hours-per-week	13.0	16.0	30.0	40.0	45.0	50.0	80.0
relationship							
Husband	1.0	NaN	NaN	2.0	1.0	NaN	1.0
Not-in-family	NaN	1.0	NaN	2.0	NaN	2.0	NaN
Own-child	NaN	NaN	1.0	NaN	NaN	NaN	NaN
Wife	NaN	NaN	NaN	2.0	NaN	NaN	NaN

[91]:
```
def reduce_hours(x):
    return x - n

data["hours-per-week"] = data["hours-per-week"].apply(reduce_hours)

Group_by_relationship_hpw = data.groupby(["relationship","hours-per-week"])

Group_by_relationship_hpw.size().unstack()
```

[91]:

hours-per-week	-18.0	-15.0	-1.0	9.0	14.0	19.0	49.0
relationship							
Husband	1.0	NaN	NaN	2.0	1.0	NaN	1.0
Not-in-family	NaN	1.0	NaN	2.0	NaN	2.0	NaN
Own-child	NaN	NaN	1.0	NaN	NaN	NaN	NaN
Wife	NaN	NaN	NaN	2.0	NaN	NaN	NaN

Lab 3

Lab Logbook Requirement:

1) Draw a bicolour features interaction diagram between the columns with the numbers of the last and second to last digits of your SID, where:

```
 #   Column
---  ------

 1   Account length
 2   Area code
 3   International plan
 4   Voice mail plan
 5   Number vmail messages
 6   Total day minutes
 7   Total day calls
 8   Total day charge
 9   Total eve minutes
 0   Total eve calls
```

In case these numbers are the same, then take the next number in order as another column number. For example, if your SID is 2287477, then you plot the bicolour diagram of the 7th and 8th columns. If your SID is 2287499, then the 9th and 0.

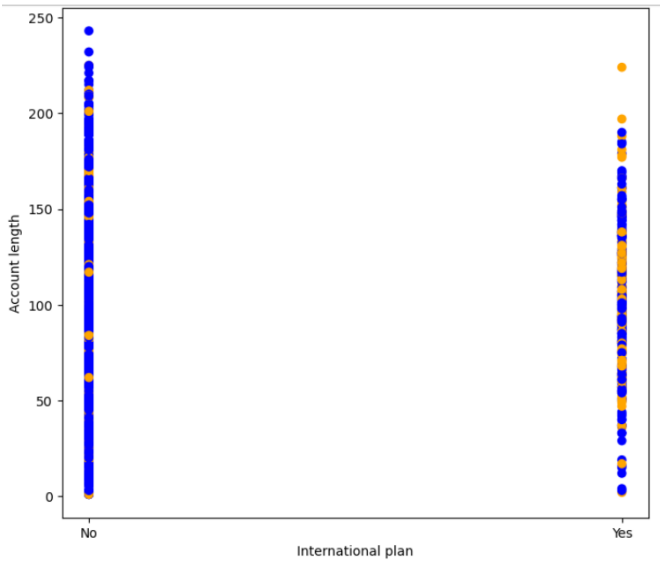
2. Add the code and result to your Lab Logbook.

NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER NOTEBOOK AND LAB LOGBOOK ON GITHUB OR ONEDRIVE.

```
Clr = data["Churn"].map({False: "blue", True: "orange"})

fig = plt.figure(figsize=(8, 7))

plt.scatter(data["International plan"], data["Account length"], color = Clr);
plt.xlabel("International plan");
plt.ylabel("Account length");
```



Lab 4

```
print(model.summary())
```

Model: "sequential_7"

Layer (type)	Output Shape	Param #
dense_25 (Dense)	(None, 531)	266,031
dense_26 (Dense)	(None, 266)	141,512
dense_27 (Dense)	(None, 1)	267

Total params: 407,810 (1.56 MB)
Trainable params: 407,810 (1.56 MB)
Non-trainable params: 0 (0.00 B)
None

My mae

```
mse, mae = model.evaluate(x_test, y_test, verbose = 0)

print("Mean absolute error: %.5f" % mae)
```

```
Mean absolute error: 0.02391
```

Practical session mae

```
mse, mae = model.evaluate(x_test, y_test, verbose = 0)

print("Mean absolute error: %.5f" % mae)
print("Mean squared error: %.5f" % mse)
```

```
Mean absolute error: 0.03089
Mean squared error: 0.00128
```

```
Mean absolute error: 0.02944
Mean squared error: 0.00125
```

Lab 5

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv1d_2 (Conv1D)	(None, 50, 50)	1,300
max_pooling1d_1 (MaxPooling1D)	(None, 7, 50)	0
conv1d_3 (Conv1D)	(None, 7, 100)	25,100
global_max_pooling1d_1 (GlobalMaxPooling1D)	(None, 100)	0
dense_2 (Dense)	(None, 25)	2,525
dense_3 (Dense)	(None, 2)	52

Total params: 28,977 (113.19 KB)
Trainable params: 28,977 (113.19 KB)
Non-trainable params: 0 (0.00 B)
None

Practical mae

```
[33]: mse, mae = model.evaluate(X_test, y_test, verbose = 1)
      print("Mean absolute error: %.5f" % mae)
```

```
936/936 ━━━━━━━━━━━━━━━━━━━━ 4s 5ms/step - loss: 0.0014 - mae: 0.0264
Mean absolute error: 0.02802
```

```
[60]:
```

```
936/936 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0013 - mae: 0.0243
Mean absolute error: 0.02579
```

My mae

```
936/936 ━━━━━━━━━━━━━━━━━━━━ 5s 5ms/step - loss: 0.0012 - mae: 0.0237
Mean absolute error: 0.02510
```

Lab 6

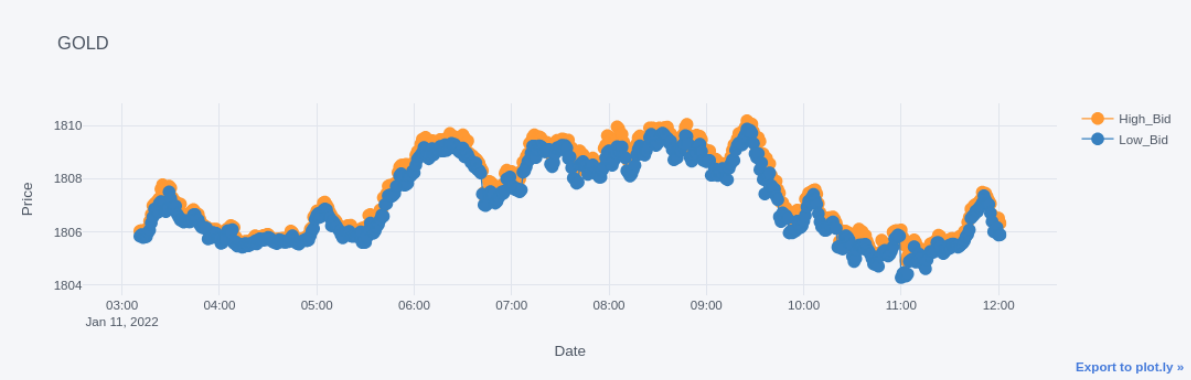
Lab Logbook Requirement:

1. Plot the price chart of the part of the whole dataset 'High_Bid' and 'Low_Bid' prices using iplot() library.
2. The start point should equal the 5 last digits of your SID Number.
3. The time period (in minutes) should equal the 3 last digits of your SID Number.
4. Please only add a print-screen of your code and final graph to your Lab Logbook.

```
<html> <h3 style="color:red;">
NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER NOTEBOOK AND LAB LOGBOOK ON GITHUB OR ONEDRIVE.
</h3> </html>
```

```
# plot the part of the whole dataset using iplot()
# The time period equals 400 minutes: from 19800 to 20200


data.iloc[8531:9062,:][['High_Bid', 'Low_Bid', 'Local_time_T', 'Volume_Ask','Volume_Bid']].iplot(
                                                x='Local_time_T', y=['High_Bid', 'Low_Bid'],
                                                mode='lines+markers',
                                                xTitle='Date', yTitle='Price',
                                                title='GOLD ')
```



Lab 7

Model: "sequential_1"

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 41)	9,840
dense_1 (Dense)	(None, 2)	84

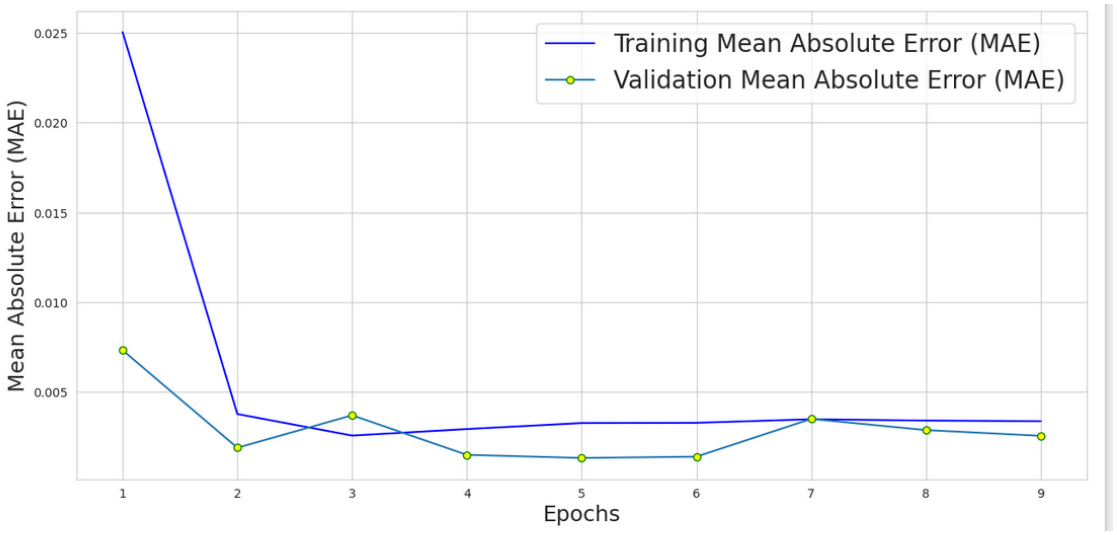
Total params: 9,924 (38.77 KB)
Trainable params: 9,924 (38.77 KB)
Non-trainable params: 0 (0.00 B)
None

```
]: print("Mean squared error (mse): %.9f " % (scores2[0]))
```

```
Mean squared error (mse): 0.000002465
```

```
]: print("Mean absolute error (mae): %.9f " % (scores2[1]))
```

```
Mean absolute error (mae): 0.001206395
```



Lab 8

Lab 9

Lab 10

Lab 11

Lab 12