LAB Logbook

Ela Dogruyol - 2351343

Lab 1

For the Lab 1 in Week 1, students were asked to create a vector using np.arange method and doing some changes on that vector to be able to practice NumPy and Python.

Firstly, because my student ID is 2351343, I created a vector of 43 elements. Secondly, I changed this matrix into a 2-d array with 1 row using **reshape** method. Thirdly, I used NumPy's **empty_like** method and **slicing** to be able to create an independent array and save the values of the matrix to that independent array. I checked the **shape** attribute values of both matrixes. I printed all results at the end of the steps.

My code and results:

Week 1 Assignment

Name: Ela Dogruyol

Student ID: 2351343

Installation process is done in the below cell.

In [1]:
```
import numpy as np
```

1) A vector that has 43 elements is created with np.arange method.

In [22]:
```
vector = np.arange(43)
print(vector)
```

```
[ 0  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 32 33 34 35 36 37 38 39 40 41 42]
```

2) Matrix is changed into a to 2-d array with 1 row.

In [23]:
```
vector = vector.reshape(43,1)
print(vector)
```

```
[[ 0]
 [ 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]
 [32]
 [33]
 [34]
 [35]
 [36]
 [37]
 [38]
 [39]
 [40]
 [41]
 [42]]
```

3) The constructed array is saved into another array.

In [24]:
```
new_array_2d = np.empty_like(vector)
new_array_2d[:,:] = vector
print(new_array_2d)
```

```
[[ 0]
 [ 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]
 [32]
 [33]
 [34]
 [35]
 [36]
 [37]
 [38]
 [39]
 [40]
 [41]
 [42]]
```

4) Shape attribute value is checked for both arrays.

```
In [26]:
print(vector.shape)
print(new_array_2d.shape)
```

```
(43, 1)
(43, 1)
```

Lab 2

For the Lab 2 in Week 2, Pandas and its main functions are studied. According to requirements, "adult_data_mini.csv" is used and some operations is done.

Firstly, n is determined as 3 (n=3) because of my student ID. Secondly, data is grouped by "relationship" and "hours-per-week". In other words, "relationship" column is grouped based on the "hours-per-week" column values. Thirdly, "hours-per-week" column values is reduced by n=3. At this step, the function "change_data(x)" is created and used. To apply this function to the dataset, apply() method is used and original DataFrame is updated. Lastly, grouping by "relationship" and reduced "hours-per-week" operation is done again.

My code and results:

1) Data is grouped by 'relationship' and 'hours-per-week'. (group 'relationship' based on 'hours-per-week')

```
In [49]:
group_by_hours = data.groupby(['relationship', 'hours-per-week'])
group_by_hours.size()
```

```
Out[49]: relationship    hours-per-week
          Husband        13.0             1
                         40.0             2
                         45.0             1
                         80.0             1
          Not-in-family  16.0             1
                         40.0             2
                         50.0             2
          Own-child      30.0             1
          Wife           40.0             2
         dtype: int64
```

2) In order to change values of the original DataFrame, a function is created. Then, all values of 'hours-per-week' is reduced by 3 (because n=3).

In [50]:
```
def change_data(x):
    return x - 3

data['hours-per-week'] = data['hours-per-week'].apply(change_data)
data
```

Out[50]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	Answer	IsHo
0	39	State-gov	77516.0	Bachelors	13.0	Never-married	Adm-clerical	Not-in-family	White	Male	2174.0	NaN	37.0	United-States	<=50K	
1	50	Self-emp-not-inc	83311.0	Bachelors	13.0	Married-civ-spouse	Exec-managerial	Husband	White	Male	0.0	0.0	10.0	United-States	<=50K	
2	38	Private	215646.0	HS-grad	9.0	Divorced	Handlers-cleaners	Not-in-family	White	Male	0.0	NaN	37.0	United-States	<=50K	
3	53	Private	234721.0	11th	7.0	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0.0	NaN	37.0	United-States	<=50K	
4	28	Private	338409.0	Bachelors	13.0	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0.0	NaN	37.0	Cuba	<=50K	
5	37	Private	284582.0	Masters	14.0	Married-civ-spouse	Exec-managerial	Wife	White	Female	0.0	NaN	37.0	United-States	<=50K	
6	49	Private	160187.0	9th	5.0	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0.0	0.0	13.0	Jamaica	<=50K	
7	52	Self-emp-not-inc	209642.0	HS-grad	9.0	Married-civ-spouse	Exec-managerial	Husband	White	Male	0.0	0.0	42.0	United-States	>50K	
8	31	Private	45781.0	Masters	14.0	Never-married	Prof-specialty	Not-in-family	White	Female	14084.0	NaN	47.0	United-States	>50K	
10	37	Private	280464.0	Some-college	10.0	Married-civ-spouse	Exec-managerial	Husband	Black	Male	0.0	NaN	77.0	United-States	>50K	
12	23	Private	122272.0	Bachelors	13.0	Never-married	Adm-clerical	Own-child	White	Female	0.0	NaN	27.0	United-States	<=50K	
13	32	Private	205019.0	Assoc-acdm	12.0	Never-married	Sales	Not-in-family	Black	Male	0.0	NaN	47.0	United-States	<=50K	
14	40	Private	121772.0	Assoc-voc	11.0	Married-civ-spouse	Craft-repair	Husband	Asian-Pac-Islander	Male	0.0	NaN	37.0	?	>50K	
15	25	Private	NaN	Some-college	NaN	NaN	NaN	NaN	White	Male	0.0	NaN	NaN	NaN	NaN	

3) Grouping is done again with 'relationship' and reduced 'hours-per-week'.

In [51]:
```
group_by_reduced_hours = data.groupby(['relationship', 'hours-per-week'])
group_by_reduced_hours.size()
```

Out[51]:
```
relationship    hours-per-week
 Husband         10.0             1
                 37.0             2
                 42.0             1
                 77.0             1
 Not-in-family   13.0             1
                 37.0             2
                 47.0             2
 Own-child       27.0             1
 Wife            37.0             2
dtype: int64
```

Lab 3

For Lab 3 in Week 3, a bicolour features interaction diagram drawing is requested as a requirement. Because of my student ID, selected columns are 3rd and 4th column for me (3-International plan, 4-Voice mail plan).

According to diagram, there is no connection between having an international plan and having a voice mail plan. Because the number of people who do not prefer the voice mail plan is approximately 2 times more than who preferred the voice mail plan for both group of people.

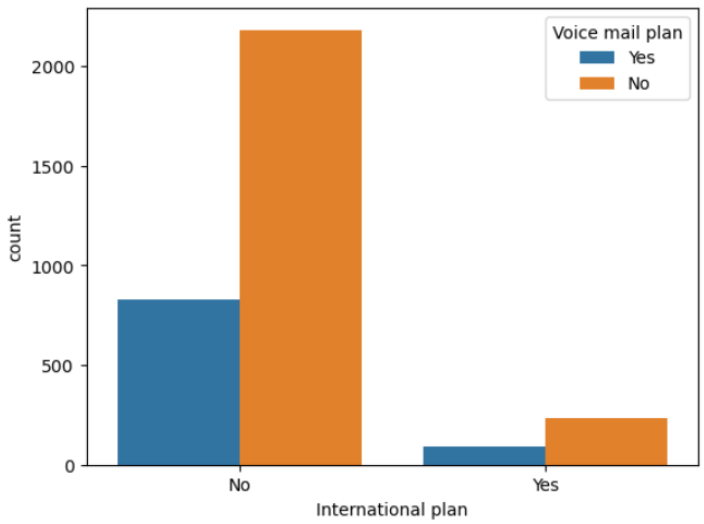
My code and result:

Name: Ela Dogruyol

Student ID: 2351343

Because of my Student ID, selected columns are 3-International plan and 4-Voice mail plan.

```
In [47]: 1 sns.countplot(x='International plan', hue='Voice mail plan', data=data);
```



According to data, whether people have international plan or not, generally they do not prefer voice mail plan. The number of people who do not prefer the voice mail plan is approximately 2 times more than who preferred the voice mail plan for both group of people.

Lab 4

For Lab 4 in Week 4, Multilayer Perceptron (MLP) Neural Network code is written in the practical session. This was our first price forecast model and S&P-500 Index Prices dataset is used. For the Lab 4, students are requested to create their own MLP model with two hidden layers.

Because of my student ID, cells inside my hidden layers are 343 for the first hidden layer and 172 for the second hidden layer. There is also an output layer with one cell. After the model creation, I compiled the model and trained it with

the same datasets of the practical session. Finally, I received the Mean Absolute Error (MAE) result.

MAE value of my model is 0.05511 while this value is 0.03434 for the MLP in the practical session. MLP in the practical session has a lower MAE value in general. MAE is a measure that is used for evaluating models and it measures the distance (difference) between the predicted and actual values for regression problems. According to this, MLP in the practical session can make more accurate predictions overall while it can be changed according to input values. MLP in the practical session has more hidden layers inside its model and it is trained with more epochs. However, the MAE value of my model is still close to the one measured for the other model, when we consider number of hidden layers and epochs.

My model summary and MAE result:

1- New MLP is created.

```
model_2 = keras.Sequential([
    keras.layers.Dense(343, input_dim=500, activation=tf.nn.relu, kernel_initializer="normal"),

    keras.layers.Dense(172, activation='relu', kernel_initializer='normal'),

    #Output layer
    keras.layers.Dense(1)
])

print(model_2.summary())
```

Model: "sequential_2"

Layer (type)	Output Shape	Param #
dense_10 (Dense)	(None, 343)	171,843
dense_11 (Dense)	(None, 172)	59,168
dense_12 (Dense)	(None, 1)	173

Total params: 231,184 (903.06 KB)

Trainable params: 231,184 (903.06 KB)

Non-trainable params: 0 (0.00 B)

None

MAE is received.

In [53]:
```
mse_2, mae_2 = model_2.evaluate(X_test, y_test, verbose=0)

print("Mean absolute error: %.5f" % mae_2)
```

Mean absolute error: 0.05511

Lab 5

For Lab 5 in Week 5, CNN code example is written in the practical session. Price forecasting for Forex EUR/USD is done. For the lab logbook requirement, another CNN model creation is requested.

In this assignment, convolutional core size should be taken as 5 and batch_size should be 50. Additionally, number of epochs should be determined accordingly to the Student IDs. Because of my student number I took number of epochs = 7 (Z=4, Y=3). After compiling and training my model, I received MAE value as a result.

Mean absolute error of the CNN in the practical session is 0.02438 while Mean absolute error of my new CNN is 0.02507. MAE shows us the average of absolute differences between actual values and predicted values. My CNN's MAE is bigger than the MAE of the CNN in the practical session. In the practical session, we used bigger kernel_size(convolutional core size) parameters for Conv1D layers inside model. Also, model is trained with 15 epochs and bacth_size=30 while I used 7 for number of epochs and 50 as batch_size. Because of the parameter differences, CNN of the practical session has a lower MAE value and hence, a better performance.

My model summary and MAE result:

1- Modifying the CNN model by taking convolutional core size = 5.

```
model_2 = keras.Sequential([
    keras.layers.Conv1D(50, 5, padding='same', input_shape=(50,5), activation=tf.nn.relu, kernel_initializer="normal"),

    keras.layers.MaxPooling1D(7),

    keras.layers.Conv1D(100, 5, padding='same', activation=tf.nn.relu, kernel_initializer="normal"),

    keras.layers.GlobalMaxPooling1D(),

    keras.layers.Dense(25, activation=tf.nn.relu, kernel_initializer="normal"),

    keras.layers.Dense(2)
])

print(model_2.summary())
```

```
Model: "sequential_2"
```

Layer (type)	Output Shape	Param #
conv1d_4 (Conv1D)	(None, 50, 50)	1,300
max_pooling1d_2 (MaxPooling1D)	(None, 7, 50)	0
conv1d_5 (Conv1D)	(None, 7, 100)	25,100
global_max_pooling1d_2 (GlobalMaxPooling1D)	(None, 100)	0
dense_4 (Dense)	(None, 25)	2,525
dense_5 (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
```

4- Receive MAE.

```
mse, mae = model_2.evaluate(X_test, y_test, verbose=1)
print("Mean absolute error: %.5f" % mae)
```

```
936/936 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - loss: 0.0012 - mae: 0.0235
Mean absolute error: 0.02507
```

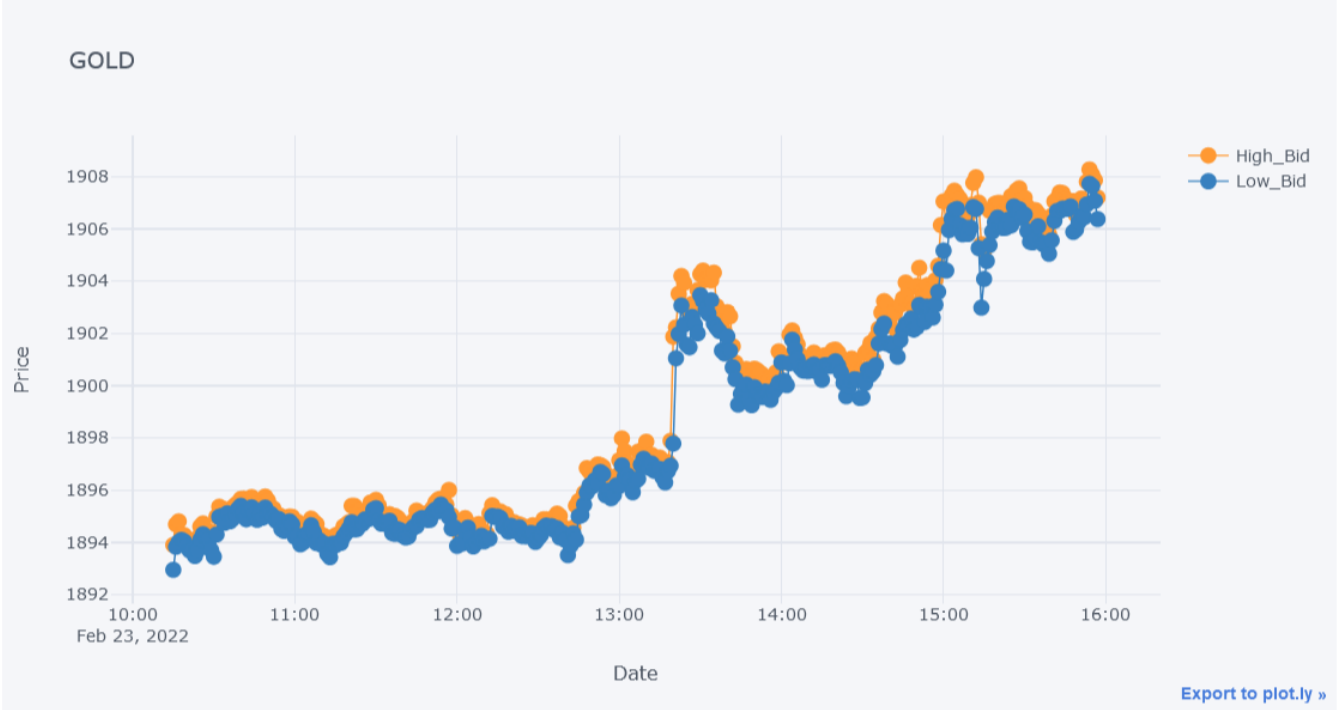
Lab 6

For Lab 6 in week 6, a detailed code of data preprocessing is written. For the Lab Logbook requirement, students were asked to draw the price chart of the part of the whole dataset 'High_Bid' and 'Low_Bid' prices using “iplot()” library.

Because of my student ID is 2351343, “start point = 51343” and “time period in minutes = 343”. In order to the time period equals to 343 minutes, data in the graph will be from 51343 to 51686 (51343 + 343 = 51686).

My code and resulting graph:

```
data.iloc[51343:51686,:][['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

For Lab 7 in Week 7, LSTM and early stopping is observed. For the Lab Logbook requirement, a new LSTM model should be conducted with the parameters as follows:

- LSTM model parameter: ZY + 10 = 43 + 10 = 53
  (Because of my Studen ID ZY = 43.)
- Epochs=10
- Patience=3

As a result, I got MAE & MSE pf my model and the model in practical session.

Practical Session:

Mean squared error (mse): 0.000002445

Mean absolute error (mae): 0.001195938

My MSE & MAE:

Mean squared error (mse): 0.000003060

Mean absolute error (mae): 0.001327695

Mean Squared Error (MSE) and Mean Absolute Error (MAE) tell us the average amount of error. Both MAE and MSE observed as smaller values in Practical session than my MAE & MSE. That means, the model constructed in practical session can predict the future values more accurate. This difference between models is because of parameter changes in LSTM model parameter, early stopping and epochs number in general.

My results:

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

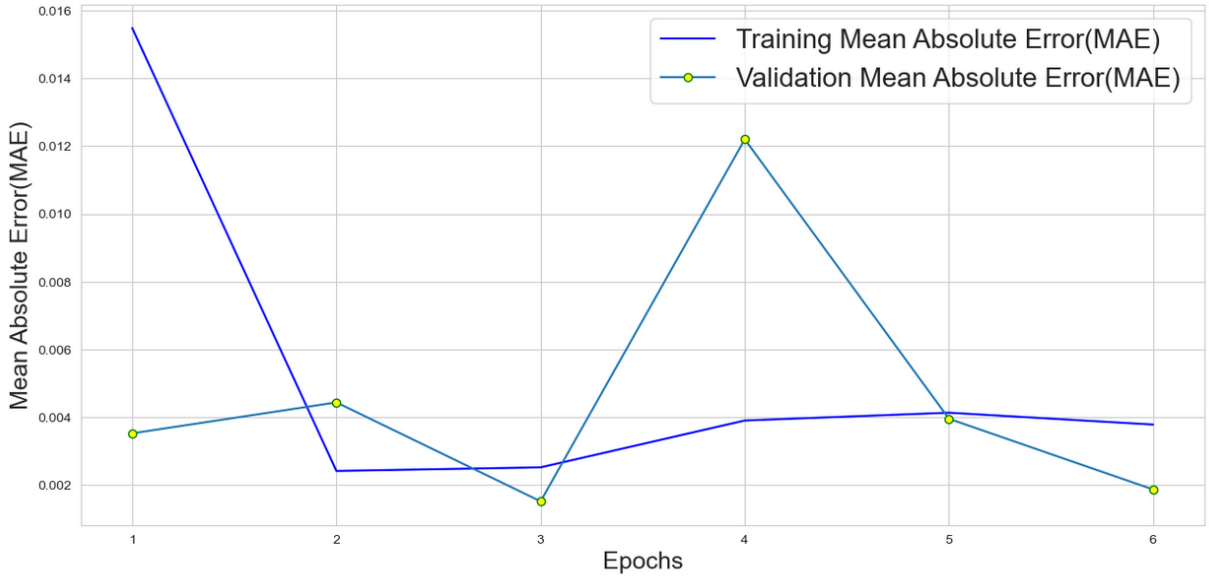
```
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 53)	15,264
dense_1 (Dense)	(None, 2)	108

```
Total params: 15,372 (60.05 KB)
Trainable params: 15,372 (60.05 KB)
Non-trainable params: 0 (0.00 B)
None
```

```
1 print("Mean squared error (mse): %.9f " % (scores[0]))
2 print("Mean absolute error (mae): %.9f " % (scores[1]))
```

```
Mean squared error (mse): 0.000003060
Mean absolute error (mae): 0.001327695
```



Lab 8

Lab 9

Lab 10

Lab 11

Lab 12