**ML in Finance Logbook**

**SID – 2322553**

**Name – Shashank Talapelliwar**

**Week 1**

**A white background with blue text

Description automatically generated**

# SID is 2322553, last two digits are 53

sid\_last\_two\_digits = 53

# Checking if the number is less than 10, if so add 100

vector\_length = sid\_last\_two\_digits if sid\_last\_two\_digits >= 10 else sid\_last\_two\_digits + 100

# Creating a vector using np.arange with the determined number of elements

vector = np.arange(vector\_length)

# Output the length of the vector and the vector itself

vector\_length, vector

**Output:-**

(53,

array([ 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, 43, 44, 45, 46, 47, 48, 49, 50,

51, 52])) 51, 52]))

a = vector

# Reshaping matrix a to a 2D array with 1 row

a\_reshaped = a.reshape(1, -1)

a\_reshaped

array([[ 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, 43, 44, 45, 46, 47,

48, 49, 50, 51, 52]])

a\_reshaped = a.reshape(1, -1)

# Saving the reshaped array into another array

b = a\_reshaped

b

array([[ 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, 43, 44, 45, 46, 47,

48, 49, 50, 51, 52]])

print("\nShape of array b:", b.shape)

Shape of array b: (1, 53)

**Github\_Url-**

**Week 2**

**A close-up of a white background

Description automatically generated**

# SID = 2322553

n = 3

data = pd.read\_csv('adult\_data\_mini.csv', header=0)

df = pd.DataFrame(data)

Group\_by\_relationship = data.groupby(['relationship', 'hours-per-week'])

print(type(Group\_by\_relationship))

Group\_by\_relationship.size()

**relationship hours-per-week**

**Husband 13 1**

**40 4**

**45 1**

**80 1**

**Not-in-family 16 1**

**40 2**

**50 2**

**Own-child 30 1**

**Wife 40 2**

**dtype: int64**

# Reducing all "hours-per-week" values by n

df['reduced-hours-per-week'] = df['hours-per-week'] - n

# Printing the updated DataFrame

print("Updated DataFrame with reduced hours-per-week:")

print(df[['hours-per-week', 'reduced-hours-per-week']])

**Updated DataFrame with reduced hours-per-week:**

**hours-per-week reduced-hours-per-week**

**0 40 37**

**1 13 10**

**2 40 37**

**3 40 37**

**4 40 37**

**5 40 37**

**6 16 13**

**7 45 42**

**8 50 47**

**9 40 37**

**10 80 77**

**11 40 37**

**12 30 27**

**13 50 47**

**14 40 37**

# Grouping by "relationship" and original "hours-per-week"

grouped\_original = df.groupby(['relationship', 'hours-per-week']).size().reset\_index(name='count')

# Grouping by "relationship" and reduced "hours-per-week"

grouped\_reduced = df.groupby(['relationship', 'reduced-hours-per-week']).size().reset\_index(name='count')

# Printing the results

print("Grouped by relationship and original hours-per-week:")

print(grouped\_original)

print("\nGrouped by relationship and reduced hours-per-week:")

print(grouped\_reduced)

**Grouped by relationship and original hours-per-week:**

**relationship hours-per-week count**

**0 Husband 13 1**

**1 Husband 40 4**

**2 Husband 45 1**

**3 Husband 80 1**

**4 Not-in-family 16 1**

**5 Not-in-family 40 2**

**6 Not-in-family 50 2**

**7 Own-child 30 1**

**8 Wife 40 2**

**Grouped by relationship and reduced hours-per-week:**

**relationship reduced-hours-per-week count**

**0 Husband 10 1**

**1 Husband 37 4**

**2 Husband 42 1**

**3 Husband 77 1**

**4 Not-in-family 13 1**

**5 Not-in-family 37 2**

**6 Not-in-family 47 2**

**7 Own-child 27 1**

**8 Wife 37 2**

**Week 3**

**A screenshot of a computer

Description automatically generated**

# SID = 2322553

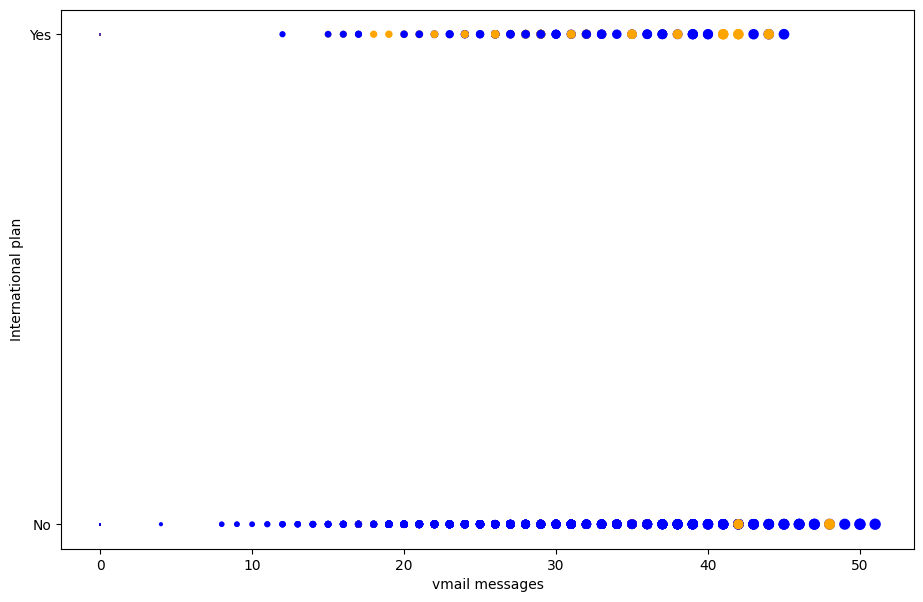
# Plotting between col 5 and 3

fig = plt.figure(figsize=(11,7))

plt.scatter(data['Number vmail messages'], data['International plan'], color = clr, s=(data['Number vmail messages']+0.05));

plt.xlabel('vmail messages');

plt.ylabel('International plan');

****

**Week 4**

**A blue text on a white background

Description automatically generated**

# SID 2322553

# Define the MLP model

model = keras.Sequential([

    keras.layers.Dense(553, input\_dim=500, activation=tf.nn.relu, kernel\_initializer="normal"),  # First hidden layer (553 neurons)

    keras.layers.Dense(277, activation='relu', kernel\_initializer="normal"),  # Second hidden layer (277 neurons)

    keras.layers.Dense(1)  # Output layer for regression (1 neuron)

])

print(model.summary())

# Compile the model

model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['mae'])

# Train the model for 10 epochs

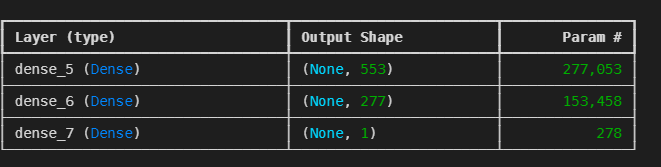
history = model.fit(X\_train,y\_train,batch\_size=10,epochs=10,validation\_split=0.2,verbose=1)

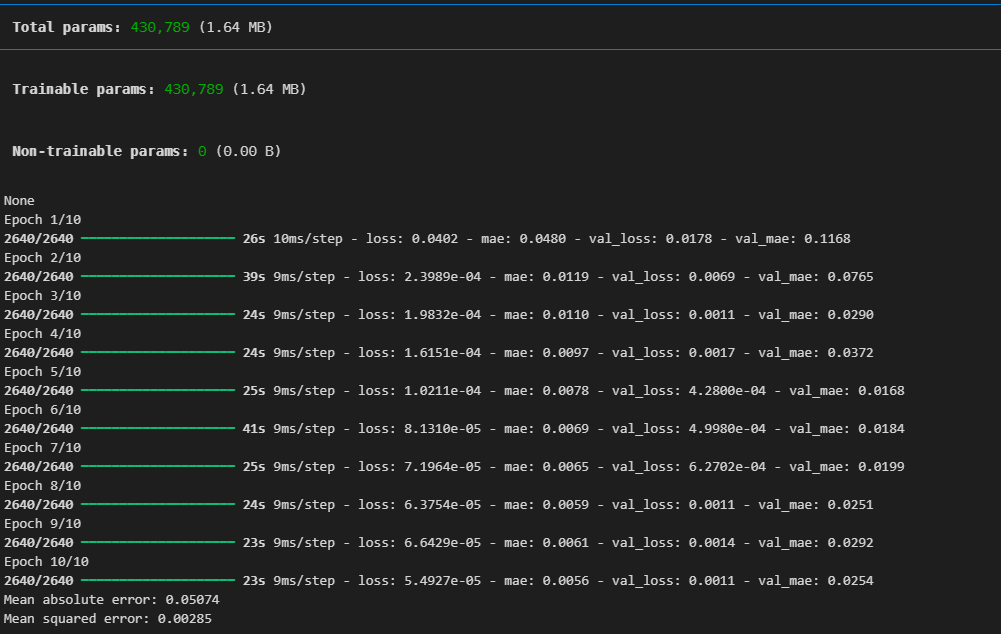
# Evaluate the model on the test data

mse, mae = model.evaluate(X\_test, y\_test, verbose=0)

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

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





**Week 5**

**1) Modify the practical session CNN model by reducing the convolutional core size to 5.**

**2) Change the batch\_size to 50.**

**3) Also, change the size of the number of epochs, which is calculated by the formula:**

**Z + Y, if Z = 0**

**10 + Y, if Z = 0 and Y is not 0**

**10, if Z = Y = 0**

**, where your SID is: XXXXXZY**

**5) Leave other parameters the same as in the practical session.**

**6) Compile the model.**

**6) Train your CNN with the same datasets and demonstrate the received test MAE.**

**Compare your MAE with the MAE of the CNN in the practical session.**

**7) Please only add a print-screen of your CNN architecture using model.summary() and the resulting MAE to your Lab Logbook.**

# CNN model with convolutional core size = 5

model = 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.summary())

