TEAM-ML_STARTERS

Problem Statement – 1:

Analyze Placement Data

TEAM MEMBERS :

1)TIRTHAJIT BORAL(LEADER)
2)SAMIRAN MAJUMDER
3)ATIF AKHTAR
4)MD TARIQUE HUSSAIN

Dataset : https://www.kaggle.com/datasets/revelation2k23/brain-dead-placement-data

Contest : Brain-Dead by Revelation '23(via Unstop)

KEY POINTS :

1)DATA CLEANING AND PROCESSING
2)DATA VISUALISATION AND ANALYSIS
3)DETAILED ANALYSIS OF PERCENTAGE ON BAR PLOTS
4)ANSWER PROBABLE QUESTIONS ON DATASET
5)RESULTS AND DISCUSSIONS

STEP 1 : DATA CLEANING AND PROCESSING

```
[83] import pandas as pd
```

DATA CLEANING AND PROECESSING

```
[84] df = pd.read_csv('/content/Placement_Data_Full_Class.csv')
```

Double-click (or enter) to edit

```
[85] df
```

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
3	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.0	Mkt&HR	59.43	Not Placed	NaN
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210	211	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	212	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	213	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	214	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0
214	215	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	NaN

214	215	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	NaN

215 rows × 15 columns

```
[98] df.drop(columns=['sl_no'],inplace = True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   gender          215 non-null    object 
 1   ssc_p           215 non-null    float64
 2   ssc_b           215 non-null    object 
 3   hsc_p           215 non-null    float64
 4   hsc_b           215 non-null    object 
 5   hsc_s           215 non-null    object 
 6   degree_p        215 non-null    float64
 7   degree_t        215 non-null    object 
 8   workex          215 non-null    object 
 9   etest_p         215 non-null    float64
 10  specialisation  215 non-null    object 
 11  mba_p           215 non-null    float64
 12  status          215 non-null    object 
 13  salary          148 non-null    float64
dtypes: float64(6), object(8)
memory usage: 23.6+ KB
```

```
[100] df.describe()
```

	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
count	215.000000	215.000000	215.000000	215.000000	215.000000	148.000000

```
] df.describe()
```

	ssc_p	hsc_p	degree_p	etest_p	mba_p	salary
count	215.000000	215.000000	215.000000	215.000000	215.000000	148.000000
mean	67.303395	66.333163	66.370186	72.100558	62.278186	288655.405405
std	10.827205	10.897509	7.358743	13.275956	5.833385	93457.452420
min	40.890000	37.000000	50.000000	50.000000	51.210000	200000.000000
25%	60.600000	60.900000	61.000000	60.000000	57.945000	240000.000000
50%	67.000000	65.000000	66.000000	71.000000	62.000000	265000.000000
75%	75.700000	73.000000	72.000000	83.500000	66.255000	300000.000000
max	89.400000	97.700000	91.000000	98.000000	77.890000	940000.000000

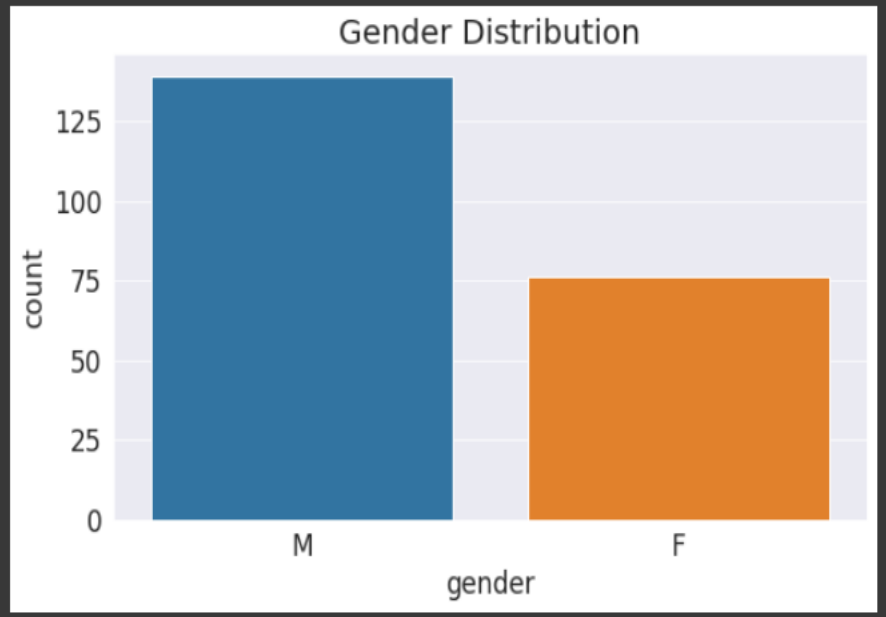
```
[101] import seaborn as sns
      import matplotlib
      import matplotlib.pyplot as plt
      %matplotlib inline

      sns.set_style('darkgrid')
      matplotlib.rcParams['font.size'] = 17
      matplotlib.rcParams['figure.figsize'] = (9, 5)
      matplotlib.rcParams['figure.facecolor'] = '#00000000'
```

STEP 2 : DATA VISUALISATION AND ANALYSIS

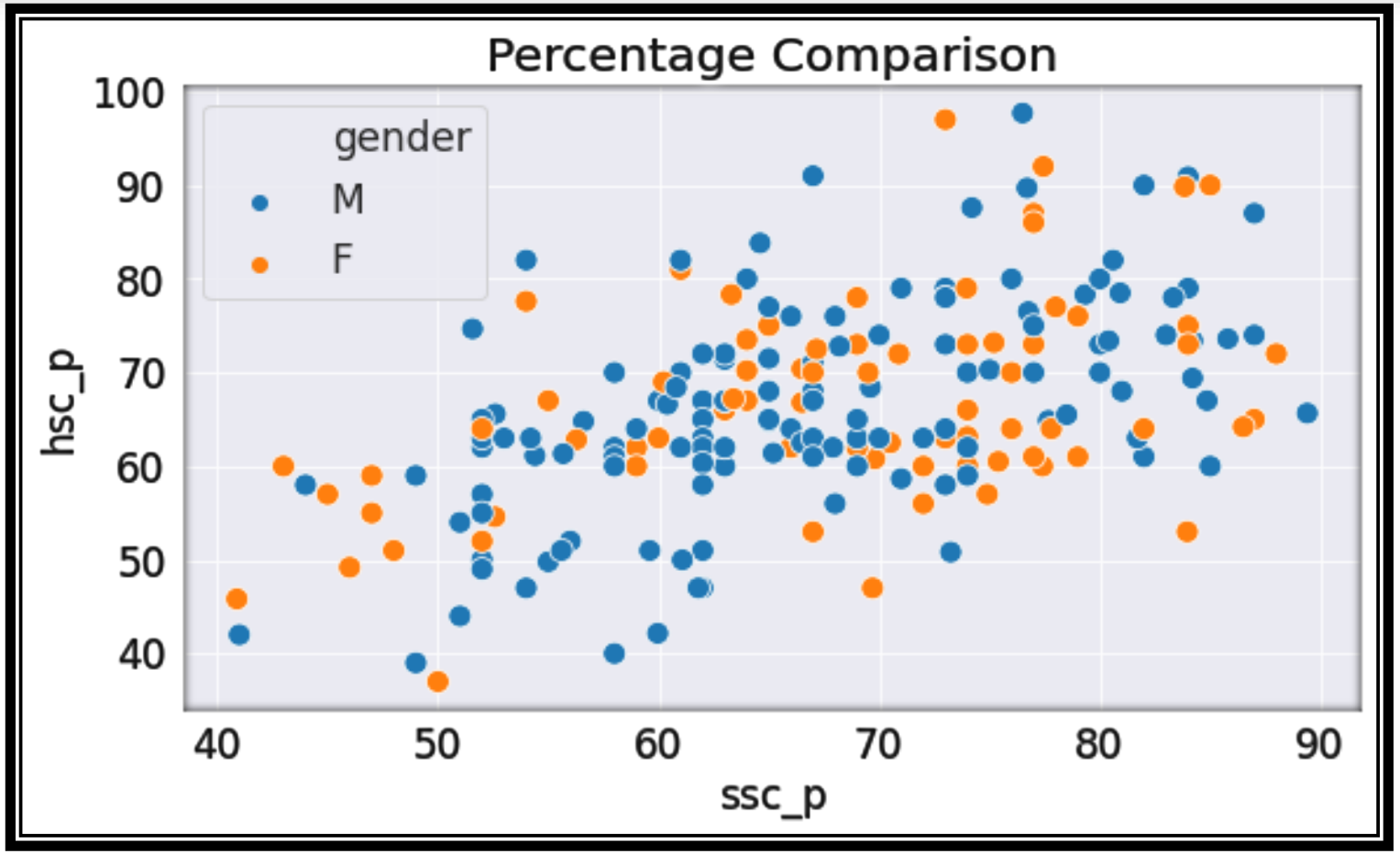
DATA VISUALISTAION AND ANALYSIS

```
print("Number of male students : ",len(df[df.gender=='M']))
print("Number of female students : ",len(df[df.gender=='F']))
sns.countplot(x = 'gender',data = df);
plt.title("Gender Distribution");
```

```
Number of male students :  139
Number of female students :  76
```



```
sns.scatterplot(df.ssc_p, df.hsc_p, hue=df.gender, s=100);
plt.title("Percentage Comparison");
```



INFERENCE : MAXIMUM OF BOYS AND GIRLS STUDENT HAS SCORED A PERCENTAGE B/W 60 TO 70 AT SEC. AND HIGHER SEC. EDUCATION.

STEP 3: DETAILED ANALYSIS OF PERCENTAGE ON BAR PLOTS

DISTRIBUTION OF SECONDARY EDUCATION PERCENTAGE AMONG MALE AND FEMALE :

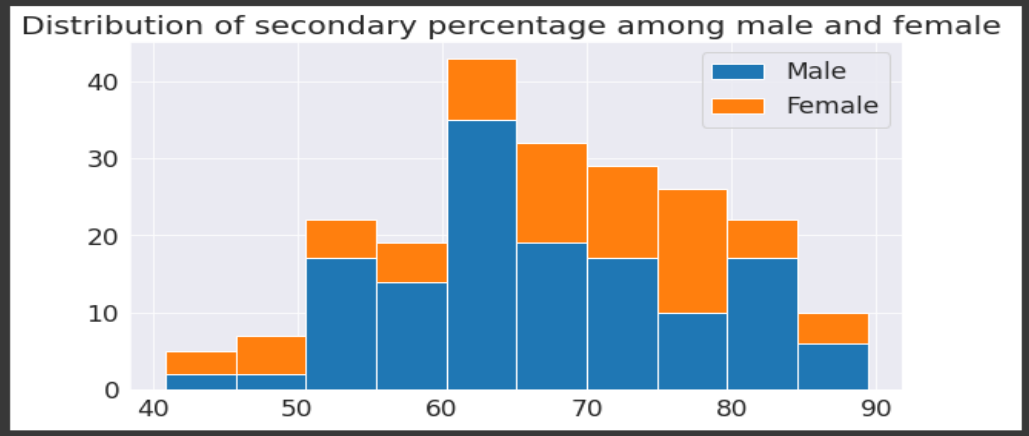
ANALYSIS ON BAR PLOTS FOR BETTER VISUALISATION OF PERCENTAGES

+ Code + Text

1) Distribution of Secondary Education percentage among male and female :

```
[109] male_df = df[df.gender == 'M']
      female_df = df[df.gender == 'F']
```

```
[110] plt.title('Distribution of secondary percentage among male and female ')

      plt.hist([male_df.ssc_p, female_df.ssc_p],
               stacked=True);

      plt.legend(['Male','Female']);
```

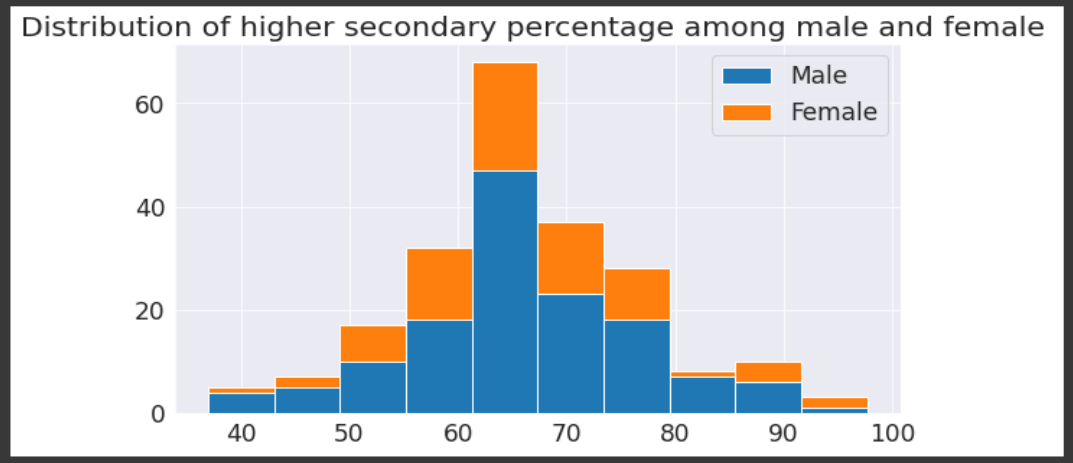


INFERENCE : MAX OF BOY STUDENTS HAS SCORED PERCENTAGE B/W 60-65 WHEREAS MAX OF GIRL STUDENT SCORED PERCENTAGE OF 75-80

DISTRIBUTION OF HIGHER SECONDARY EDUCATION PERCENTAGE AMONG MALE AND FEMALE:

2) Distribution of Higher Secondary Education percentage among male and female :

```
plt.title('Distribution of higher secondary percentage among male and female ')

plt.hist([male_df.hsc_p, female_df.hsc_p],
         stacked=True);

plt.legend(['Male','Female']);
```

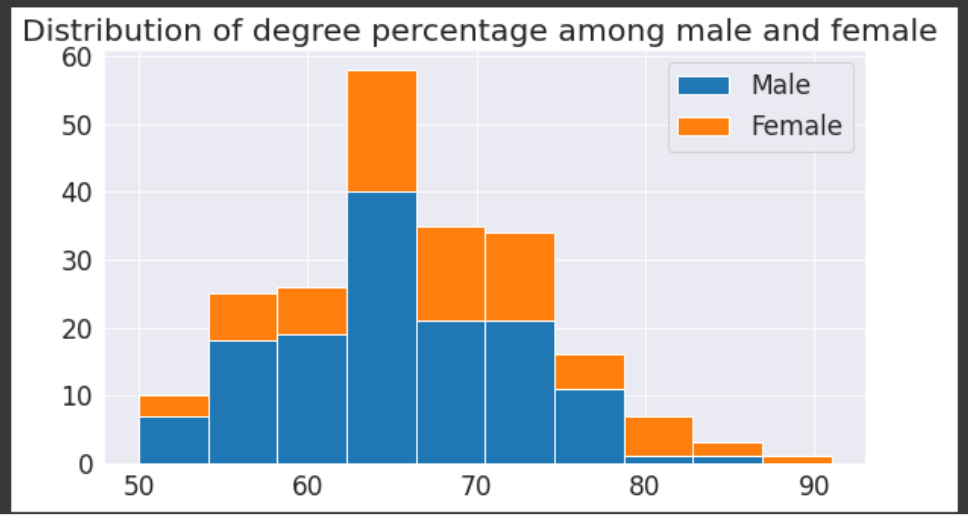


INFERENCE : MAX OF BOY STUDENTS HAS SCORED PERCENTAGE B/W 60-70 SAME FOR GIRL STUDENTS ALSO

DISTRIBUTION OF DEGREE PERCENTAGE AMONG MALE AND FEMALE :

5) Distribution of Degree percentage among male and female :

```
plt.title('Distribution of degree percentage among male and female ')

plt.hist([male_df.degree_p, female_df.degree_p],stacked='True');

plt.legend(['Male','Female']);
```

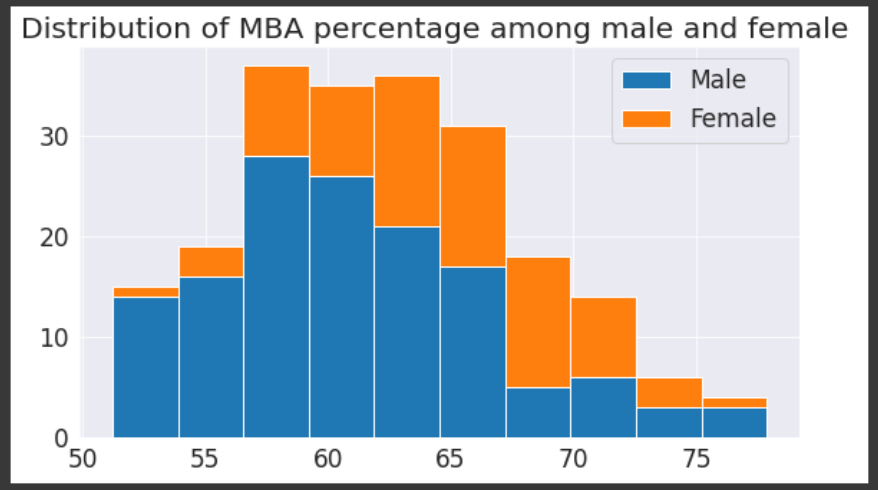


INFERENCE : MAX OF BOY STUDENTS HAS SCORED PERCENTAGE B/W 60-70 SAME FOR GIRL STUDENTS ALSO.

DISTRIBUTION OF MBA PERCENTAGE AMONG MALE AND FEMALE:

4)Distribution of MBA percentage among male and female :

```
plt.title('Distribution of MBA percentage among male and female ')

plt.hist([male_df.mba_p, female_df.mba_p],
         stacked=True);

plt.legend(['Male','Female']);
```



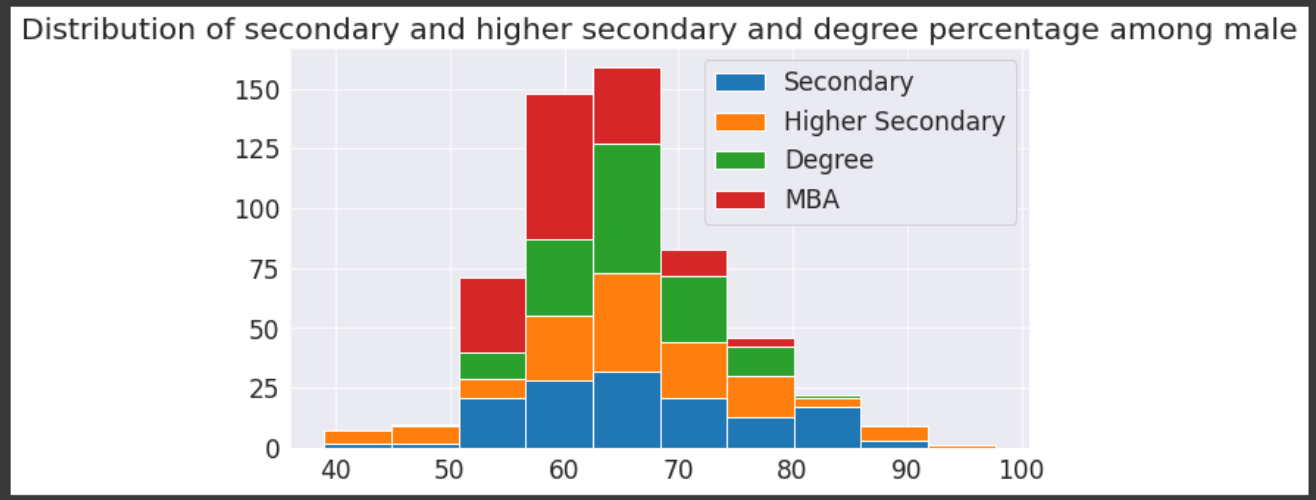
Inference : Max no. of boys student has scored a percentage of 55-60 during MBA studies while max of girls students scored a percentage of 60-65

INFERENCE : MAX NO. OF BOYS STUDENT HAS SCORED A PERCENTAGE OF 55-60 DURING MBA STUDIES WHILE MAX OF GIRLS STUDENTS SCORED A PERCENTAGE OF 60-65

DISTRIBUTION OF SECONDARY , HIGHER SECONDARY , DEGREE PERCENTAGE AMONGST MALE STUDENTS:

)Distribution of secondary , higher secondary , degree percentage amongst male students:

```
plt.title('Distribution of secondary and higher secondary and degree percentage among male')

plt.hist([male_df.ssc_p, male_df.hsc_p, male_df.degree_p,male_df.mba_p],
         stacked=True);

plt.legend(['Secondary','Higher Secondary','Degree','MBA']);
```



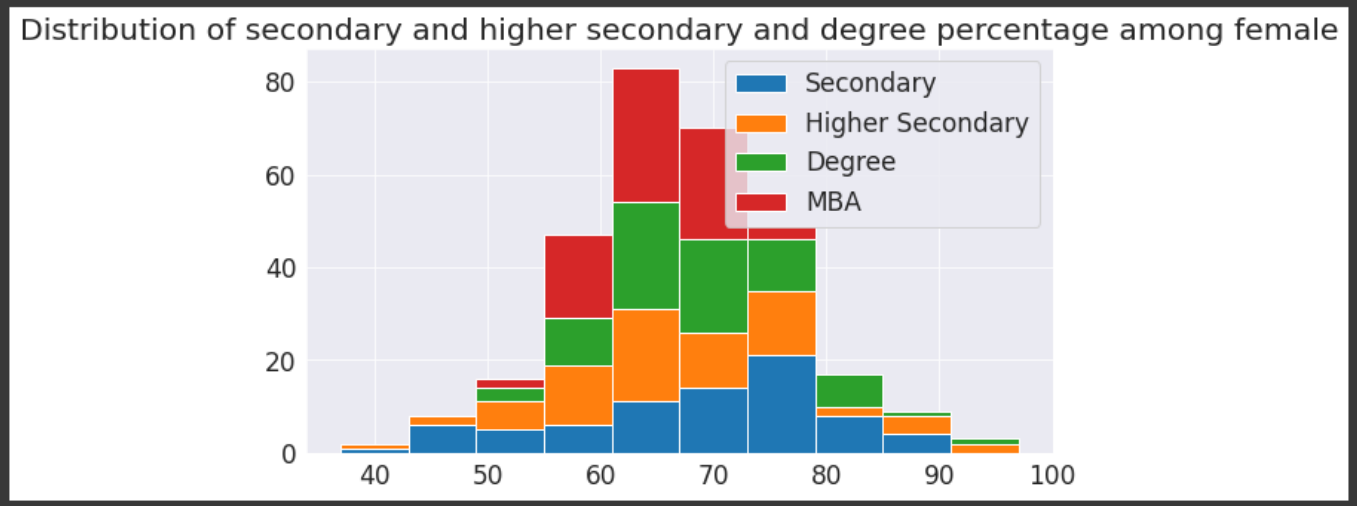
Inference : Boy students maintained their percentage of 60-70 till degree studies and after that it has started dropping during mba studies

INFERENCE : BOY STUDENTS MAINTAINED THEIR PERCENTAGE OF 60-70 TILL DEGREE STUDIES AND AFTER THAT IT HAS STARTED DROPPING DURING MBA STUDIES

DISTRIBUTION OF SECONDARY , HIGHER SECONDARY , DEGREE PERCENTAGE AMONGST FEMALE STUDENTS:

5)Distribution of secondary , higher secondary , degree percentage amongst female students:

```
plt.title('Distribution of secondary and higher secondary and degree percentage among female')

plt.hist([female_df.ssc_p, female_df.hsc_p, female_df.degree_p,female_df.mba_p],
         stacked=True);

plt.legend(['Secondary','Higher Secondary','Degree','MBA']);
```



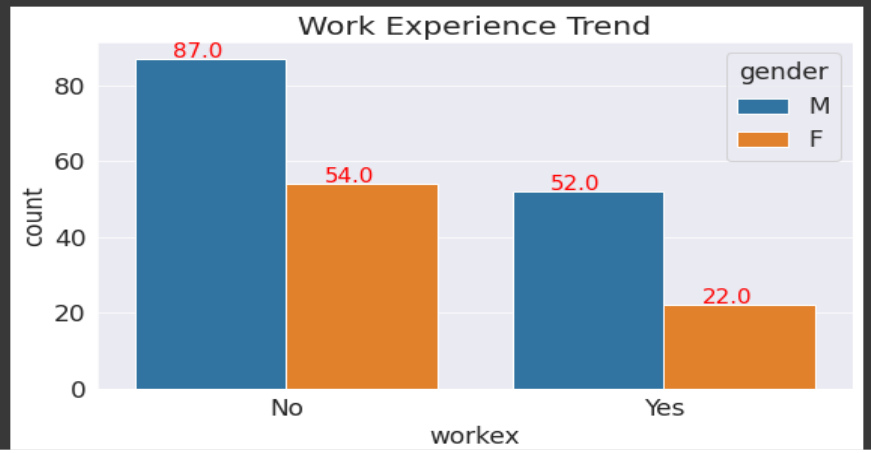
Inference : Girl students scored the maximum marks during secondary studies which is 70-80 which dropped down to 60-70 and it is maintained till degree studies

INFERENCE : GIRL STUDENTS SCORED THE MAXIMUM MARKS DURING SECONDARY STUDIES WHICH IS 70-80 WHICH DROPPED DOWN TO 60-70 AND IT IS MAINTAINED TILL DEGREE STUDIES

6)ANALYIS OF WORK EXPERIENCE TREND :

6)Analyis of work experience trend :

```
ax=sns.countplot(x='workex', hue=df.gender, data=df);
totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)


for i in ax.patches:
    ax.text(i.get_x()-.1, i.get_height()+.5, \
            str(round(i.get_height(), 2)), fontsize=15,
                color='red')

plt.title("Work Experience Trend");
```



INFERENCE : 52 OUT OF 139 BOYS HAVE WORK EXPIRIENCE AND ONLY 22 OUT OF 76 GIRLS HAVE WORK EXPERIENCE

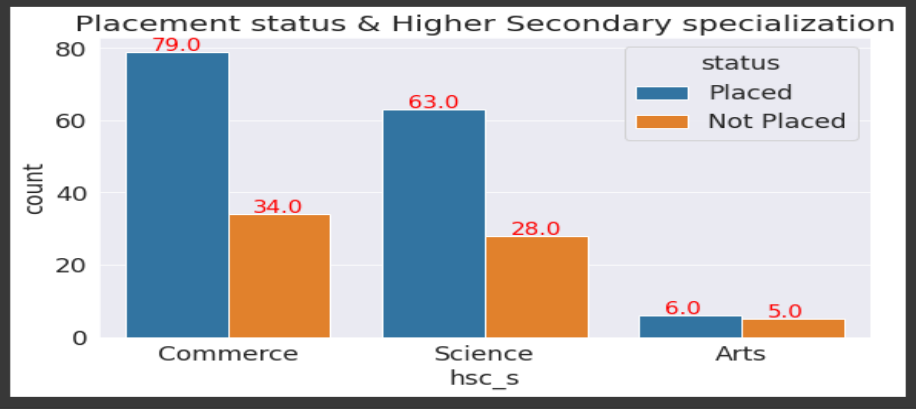
STEP 4 : PROBABLE QUESTIONS ON DATASET

Q1) WHICH HIGHER SECONDARY SPECIALISATION IS ON HIGH DEMAND?

NSWERING FEW QUESTIONS ON THIS PLACEMENT DATA ANALYSIS PROBLEM STATEMENT

Q1) Which Higher secondary specialistaion is on high demand ?

```
ax=sns.countplot(x='hsc_s', hue=df.status, data=df);
totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)


for i in ax.patches:

    ax.text(i.get_x()-.1, i.get_height()+.5, \
            str(round(i.get_height(), 2)), fontsize=15,
                color='red')

plt.title("Placement status & Higher Secondary specialization");
```



INFERENCE : FROM BAR CHART PLOT IT IS CLEAR THAT SPECIALISATION IN “COMMERCE” IS ON HIGH DEMAND IN TERMS OF PLACEMENT

Q2) DOES MBA PERCENTAGE AFFECT PLACEMENT ?

Q2)Does MBA percentage affect placement ?

```
[ ]  not_placed= df[df.status=='Not Placed']
     print("Minimum percentage scored in MBA : ",df.mba_p.min())
     print("Maximum percentage scored in MBA : ",df.mba_p.max())
     print("Minimum percentage scored in MBA by students who were not placed : ",not_placed.mba_p.min())
     print("Maximum percentage scored in MBA by students who were not placed : ",not_placed.mba_p.max())
```

```
Minimum percentage scored in MBA :  51.21
Maximum percentage scored in MBA :  77.89
Minimum percentage scored in MBA by students who were not placed :  51.21
Maximum percentage scored in MBA by students who were not placed :  75.71
```

```
[ ]  df[df.mba_p==not_placed.mba_p.max()]
```

	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
167	M	67.9	Others	62.0	Others	Science	67.0	Sci&Tech	Yes	58.1	Mkt&Fin	75.71	Not Placed	NaN

Inference : MBA percentage dont have a major effect on placements acc to the stats.

INFERENCE : MBA PERCENTAGE DON'T HAVE A MAJOR EFFECT ON PLACEMENT

Q3) COMMENT ON MAXIMUM , AVERAGE , MINIMUM SALARY OFFERED ?

Q3)Comment on max , avg , min salary offered ?

```
print("Maximum salary offered",df.salary.max())
print("Minimum salary offered",df.salary.min())
print("Average salary offered",df.salary.mean())
```

```
Maximum salary offered 940000.0
Minimum salary offered 200000.0
Average salary offered 288655.4054054054
```

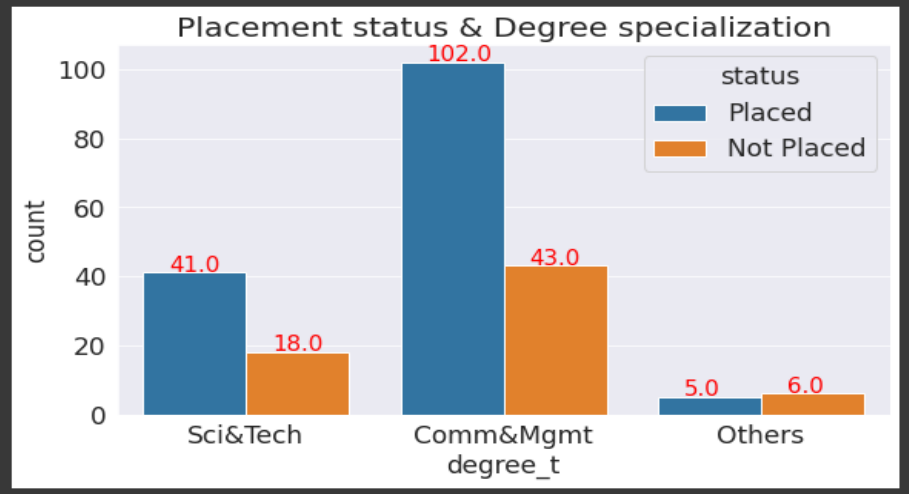
Q4) WHICH DEGREE SPECIALISTAION IS ON HIGH DEMAND ?

Q4)Which degree specialistaion is on high demand ?

```
ax=sns.countplot(x='degree_t', hue=df.status, data=df);
totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:

    ax.text(i.get_x()-.1, i.get_height()+.5, \
            str(round(i.get_height(), 2)), fontsize=15,
                color='red')
plt.title("Placement status & Degree specialization");
```

Which degree specialistaion is on high demand ?

```
ax=sns.countplot(x='degree_t', hue=df.status, data=df);
totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:

    ax.text(i.get_x()-.1, i.get_height()+.5, \
            str(round(i.get_height(), 2)), fontsize=15,
                color='red')
plt.title("Placement status & Degree specialization");
```



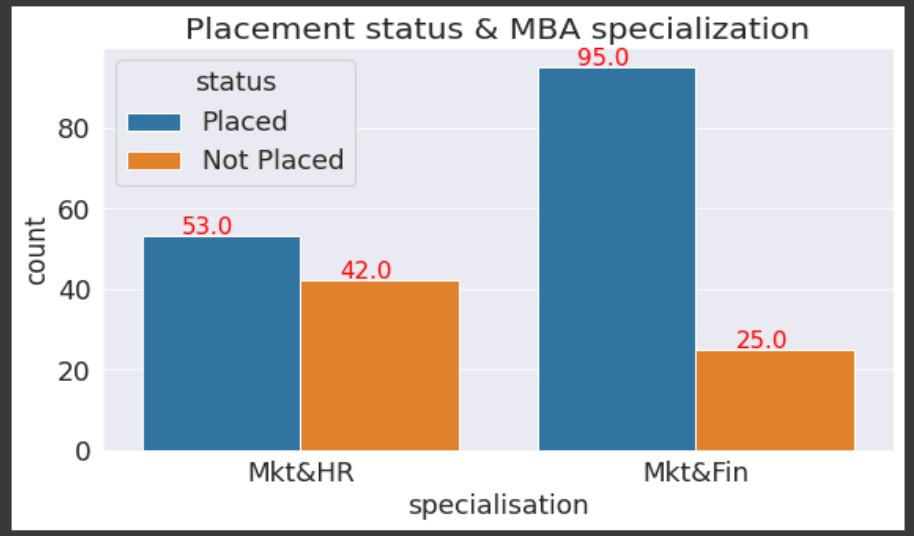
INFERENCE : From above bar chart plot it is clear that specilsation of "Management Degree" is on high demand in terms of placement for both boys and girls.

INFERENCE : FROM ABOVE BAR CHART PLOT IT IS CLEAR THAT SPECIALISATION ON "MANAGEMENT DEGREE" IS ON HIGH DEMAND IN TERMS OF PLACEMENT FOR BOTH BOYS AND GIRLS

5) CALCULATE THE NUMBER OF STUDENTS WHO GOT PLACED DEPENDING ON MBA SPECILISATION

alculate tthe number of students who got placed depending on MBA Specilisation?

```
ax=sns.countplot(x='specialisation', hue=df.status, data=df);
totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:

    ax.text(i.get_x()-.1, i.get_height()+.5, \
            str(round(i.get_height(), 2)), fontsize=15,
                color='red')
plt.title("Placement status & MBA specialization");
```



Inference : Marketing and Finance is on high demand in terms of placement

INFERENCE : MARKETING AND FINANCE IS ON HIGH DEMAND IN ACCOUNT OF PLACEMENT

STEP 5 : RESULTS AND DISCUSSIONS

So from the analysis that we have made we can conclude that:

1)The placement percentage of the campus is 68.8, which is not bad and the maximum salary offered is 9,40,000 and 24% of students were offered more than average salary of 2,88,655.

2)MBA percentage does not have much effect on getting placed. As we saw, even a student with high percentage as 75.71 and having work experience is not placed. Having a work experience will increase your chance of getting placed. We saw that, out of 74 students who had work experience 64 of them got placed in comparison to students who didn’t have a work experience.

3)We can also predict what type of specialization in MBA is preferred by employers, Marketing and Finance. Out of 120 students who specialized in Marketing and Finance, 95 of them were placed whereas in Marketing and HR, its almost a 50-50 chance of placement.

4)We can also see that the degree specialization also affects employment. Employers prefer those who have studied Commerce and Management degree.

Therefore we can conclude that, in order to achieve a management job with better salary offers, its better to choose Commerce and Management for degree specialization and Marketing and Finance for MBA specialization along with a work experience

TEAM-ML_STARTERS

PROBLEM STATEMENT-2

Detecting Emotional Sentiment in Cartoons

Data-set: https://www.kaggle.com/datasets/revelation2k23/brain-dead-emotion-detection

STEP-1:IMPORTING LIBRARIES

STEP-2:DATA EXPLORATION

Assigning Path for Dataset

```
[6]  TRAIN_DIR = "/content/drive/MyDrive/Train"
     TEST_DIR = "/content/drive/MyDrive/Test"
     BATCH_SIZE=64
```

Will see how many categories and images present

```
[7]  for folder in os.listdir(TRAIN_DIR):
         files = gb.glob(pathname= str(TRAIN_DIR+ '/'+ folder + '/*.png'))
         print(f'For training data, found {len(files)} in folder {folder}')
```

```
For training data, found 408 in folder angry
For training data, found 595 in folder happy
For training data, found 399 in folder neutral
For training data, found 408 in folder sad
```

```
[8]  for folder in os.listdir(TEST_DIR):
         files = gb.glob(pathname= str(TEST_DIR+ '/'+ folder + '/*.png'))
         print(f'For testing data, found {len(files)} in folder {folder}')
```

```
For testing data, found 105 in folder happy
For testing data, found 102 in folder neutral
For testing data, found 82 in folder sad
For testing data, found 80 in folder angry
```

STEP-2:DATA EXPLORATION

Will see some random images with their emotion classes

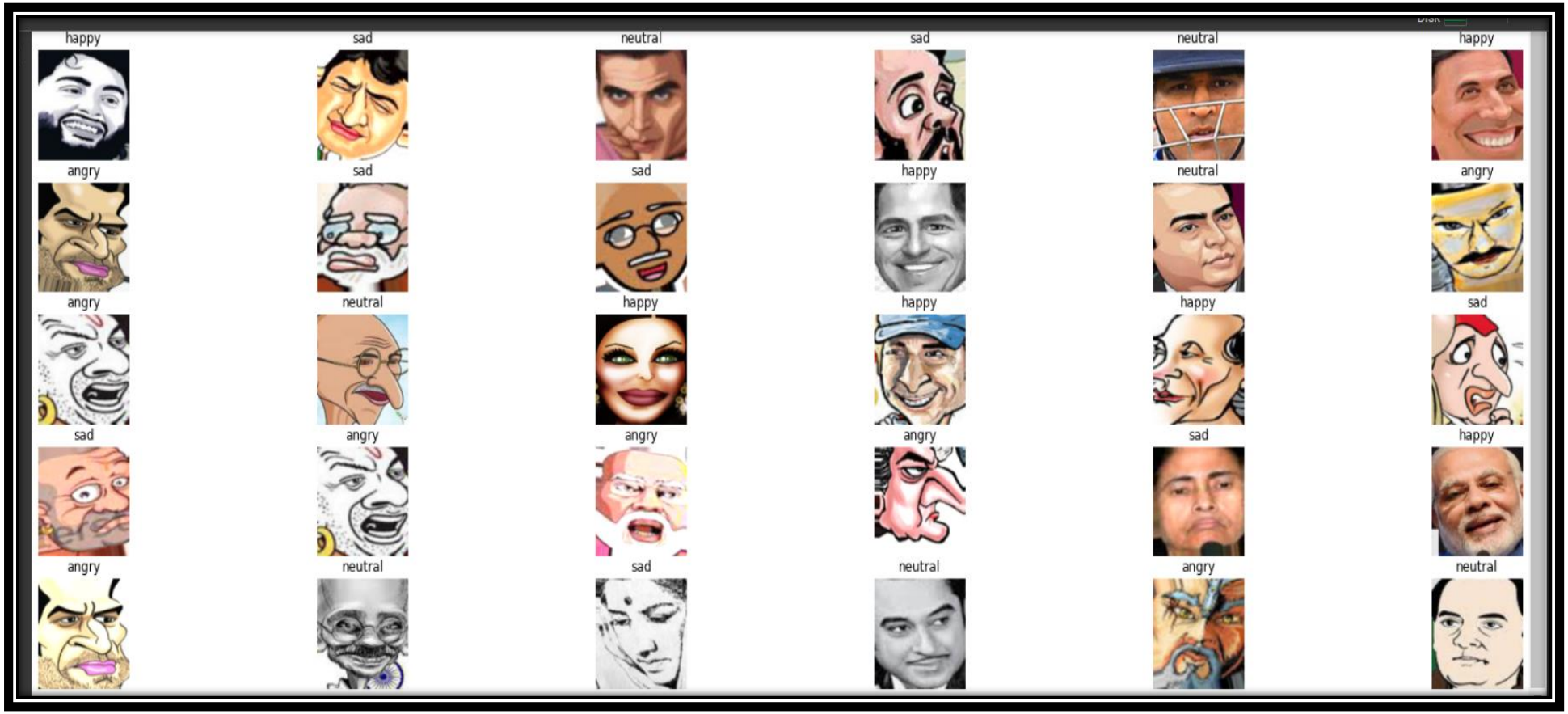
```
[49] import random
     import matplotlib.pyplot as plt
     import matplotlib.image as mpimg

     def view_random_image(target_dir, target_class):
         target_folder = target_dir + target_class

         random_image = random.sample(os.listdir(target_folder), 1)

         img = mpimg.imread(target_folder+'/'+random_image[0])
         plt.imshow(img)
         plt.title(target_class)
         plt.axis('off')
         print(f"Image shape {img.shape}")

         return img
```

```
[10] class_names = ['angry','happy','neutral','sad']
```

```
plt.figure(figsize=(30,10))
for i in range(30):
    plt.subplot(5, 6, i+1)
    class_name = random.choice(class_names)
    img = view_random_image(target_dir="/content/drive/MyDrive/Train/", target_class=class_name)
```

0s completed at 10:04 PM

STEP-2:DATA EXPLORATION (RANDOM IMAGES CLASSIFICATION)



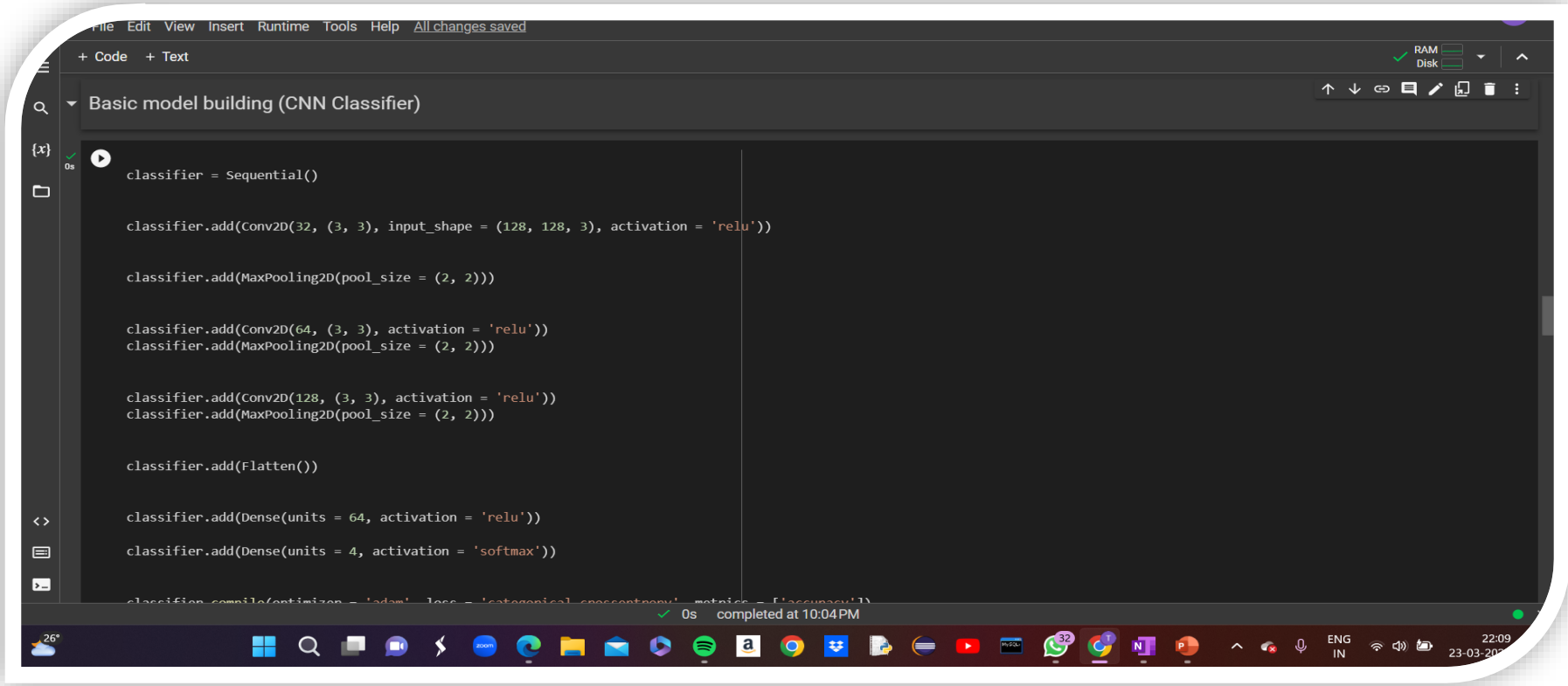
STEP-3:DATA TRAINING OF MODEL

Preparing data for training

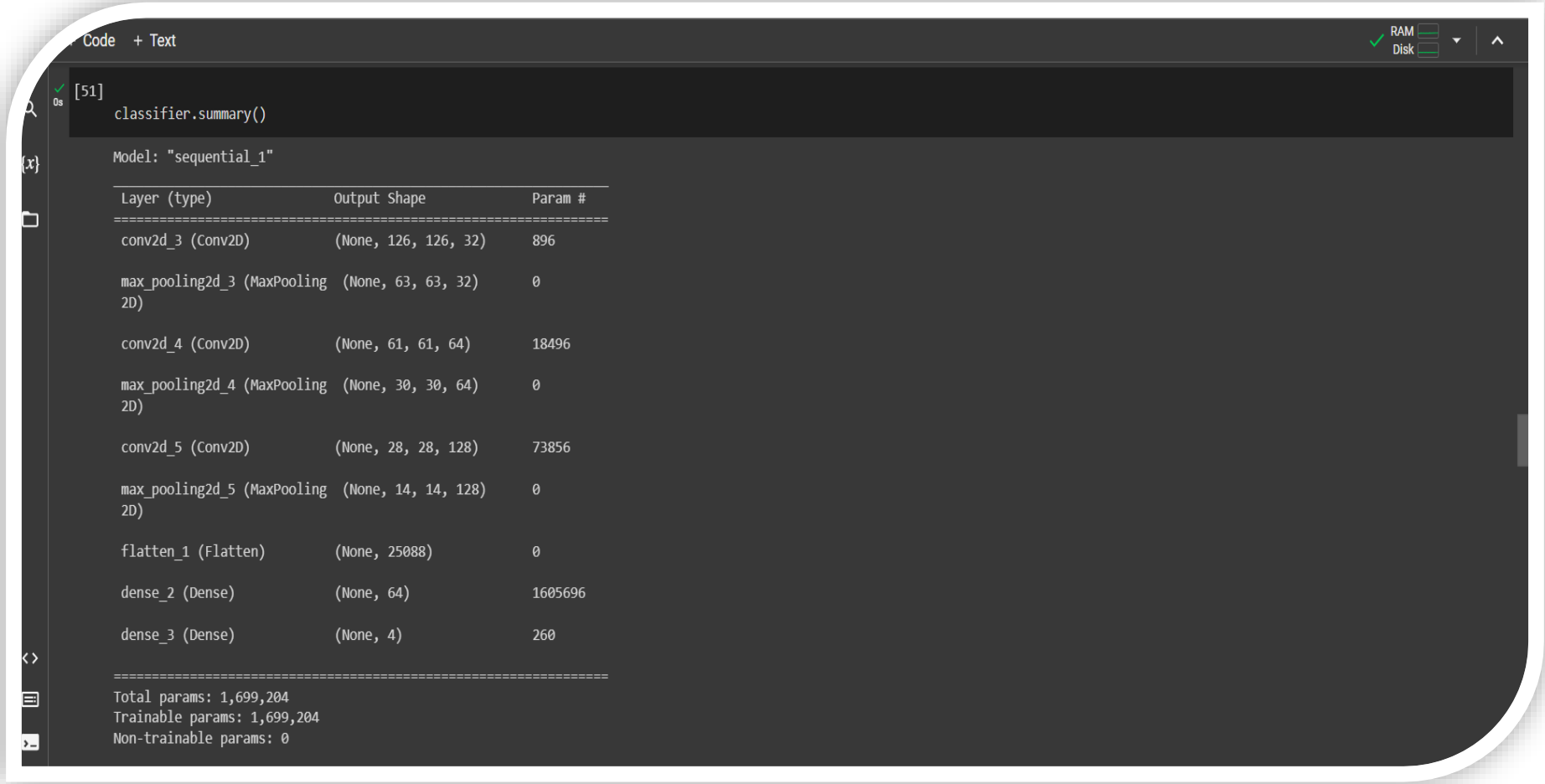
```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,
                                  zoom_range = 0.2,
                                  horizontal_flip = True)

training_set = train_datagen.flow_from_directory(TRAIN_DIR,
                                                 target_size = (128, 128),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')

test_set = test_datagen.flow_from_directory(TEST_DIR,
                                            target_size = (128, 128),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 1810 images belonging to 4 classes.
Found 369 images belonging to 4 classes.
```

STEP-4:SEQUENTIAL CNN MODEL



STEP-4:SEQUENTIAL CNN MODEL(SUMMARY)



+ Code + Text

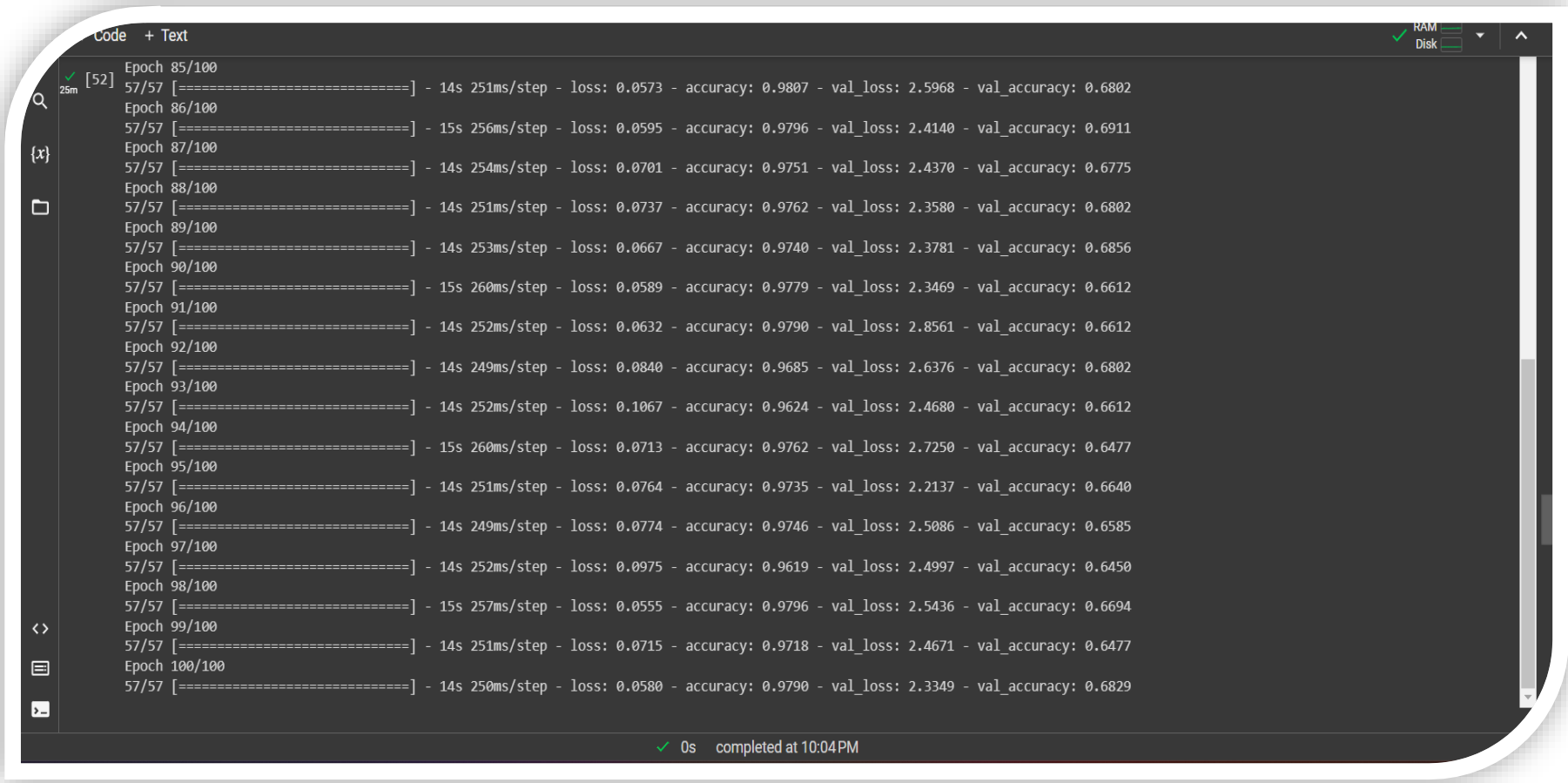
RAM Disk

[51]
0s

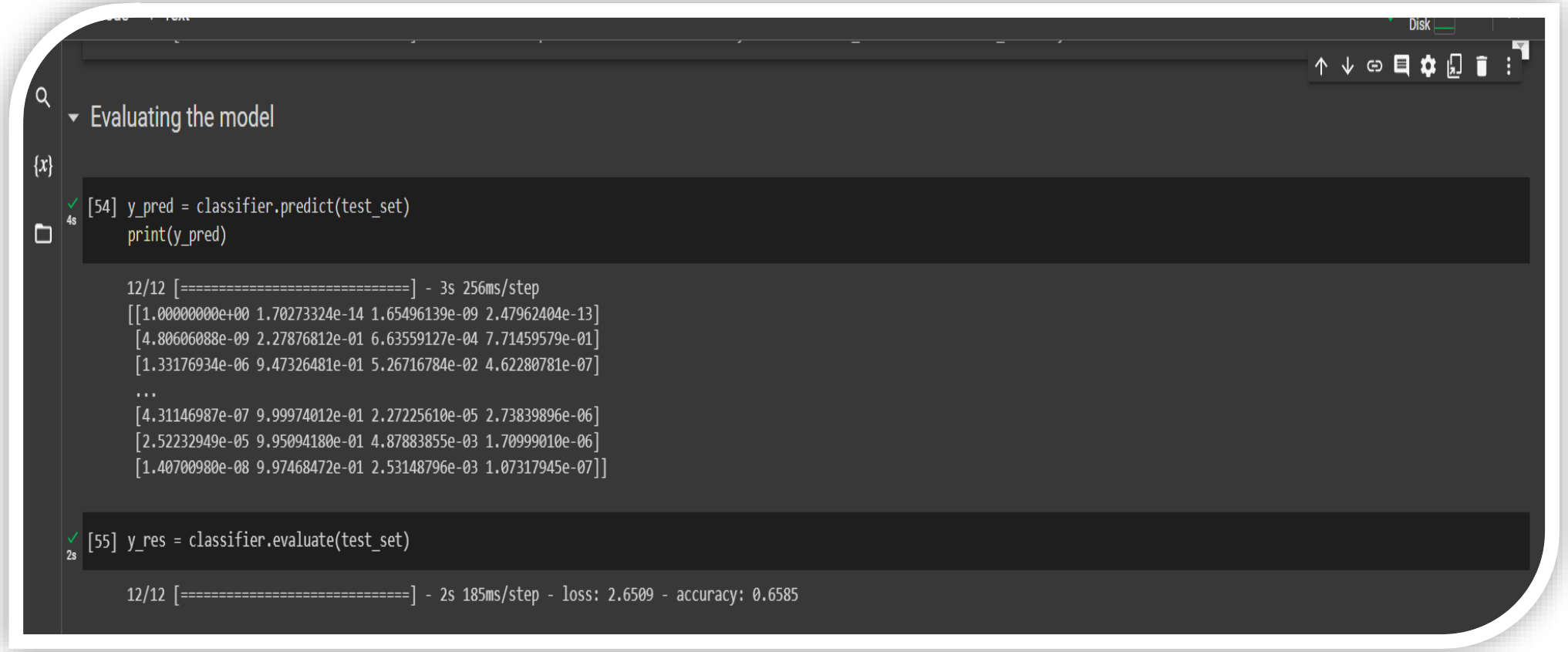
```
classifier.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 126, 126, 32)      896

 max_pooling2d_3 (MaxPooling  (None, 63, 63, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 61, 61, 64)        18496

 max_pooling2d_4 (MaxPooling  (None, 30, 30, 64)       0
 2D)

 conv2d_5 (Conv2D)           (None, 28, 28, 128)       73856

 max_pooling2d_5 (MaxPooling  (None, 14, 14, 128)      0
 2D)

 flatten_1 (Flatten)         (None, 25088)             0

 dense_2 (Dense)             (None, 64)                1605696

 dense_3 (Dense)             (None, 4)                 260

=================================================================
Total params: 1,699,204
Trainable params: 1,699,204
Non-trainable params: 0
```

TRAINING IN PROGRESS.....



STEP-5:SEQUENTIAL MODEL EVALUATION

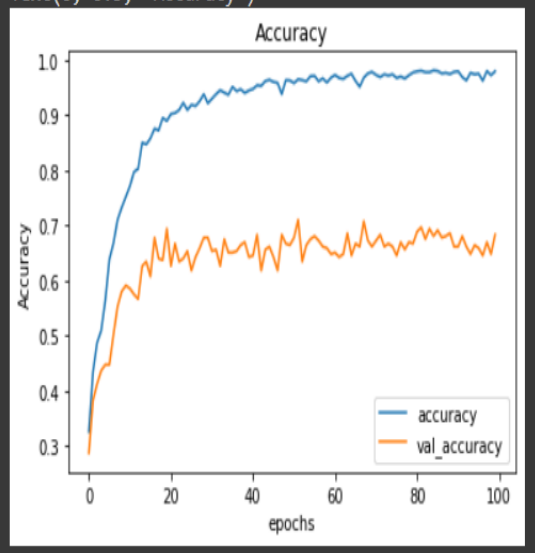


STEP-5:SEQUENTIAL MODEL EVALUATION

(PLOT SHOWING ACCURACY V/S EPOCH)

```
[56] pd.DataFrame(history.history)[['accuracy','val_accuracy']].plot()
     plt.title('Accuracy')
     plt.xlabel('epochs')
     plt.ylabel('Accuracy')
```

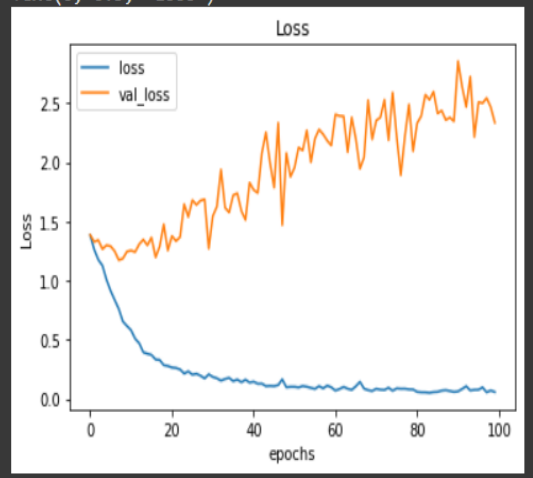
```
Text(0, 0.5, 'Accuracy')
```



STEP-5:SEQUENTIAL MODEL EVALUATION (PLOT SHOWING LOSS V/S EPOCH)

```
[57] pd.DataFrame(history.history)[['loss','val_loss']].plot()
     plt.title('Loss')
     plt.xlabel('epochs')
     plt.ylabel('Loss')
```

```
Text(0, 0.5, 'Loss')
```



STEP-6:REAL WORLD PREDICTION(ANGRY)

Prediction with new images

```
model_path = "model1.h5"
loaded_model = keras.models.load_model(model_path)


image = cv2.imread("/content/drive/MyDrive/Test/angry/Mahatma_Gandhi61.png")

image_fromarray = Image.fromarray(image, 'RGB')
resize_image = image_fromarray.resize((128, 128))
expand_input = np.expand_dims(resize_image,axis=0)
input_data = np.array(expand_input)
input_data = input_data/255

pred = loaded_model.predict(input_data)
result = pred.argmax()
print("Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:->",result)
```

```
1/1 [==============================] - 0s 75ms/step
Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:-> 0
```

```
[61] training_set.class_indices
```

```
{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}
```

STEP-6:REAL WORLD PREDICTION(HAPPY)

Prediction with new images

```
model_path = "model1.h5"
loaded_model = keras.models.load_model(model_path)


image = cv2.imread("/content/drive/MyDrive/Test/happy/Sachin_Tendulkar84.png")

image_fromarray = Image.fromarray(image, 'RGB')
resize_image = image_fromarray.resize((128, 128))
expand_input = np.expand_dims(resize_image,axis=0)
input_data = np.array(expand_input)
input_data = input_data/255

pred = loaded_model.predict(input_data)
result = pred.argmax()
print("Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:->",result)
```

```
1/1 [==============================] - 0s 74ms/step
Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:-> 1
```

```
[61] training_set.class_indices
```

```
{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}
```

STEP-6:REAL WORLD PREDICTION(NEUTRAL)

Prediction with new images

```
model_path = "model1.h5"
loaded_model = keras.models.load_model(model_path)


image = cv2.imread("/content/drive/MyDrive/Test/neutral/Shahrukh_Khan134.png")

image_fromarray = Image.fromarray(image, 'RGB')
resize_image = image_fromarray.resize((128, 128))
expand_input = np.expand_dims(resize_image,axis=0)
input_data = np.array(expand_input)
input_data = input_data/255

pred = loaded_model.predict(input_data)
result = pred.argmax()
print("Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:->",result)
```

```
1/1 [==============================] - 0s 115ms/step
Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:-> 2
```

```
training_set.class_indices
```

```
{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}
```

```
[ ]
```

1s completed at 10:46 PM

STEP-6:REAL WORLD PREDICTION(SAD)

Prediction with new images

```
model_path = "model1.h5"
loaded_model = keras.models.load_model(model_path)


image = cv2.imread("/content/drive/MyDrive/Test/sad/Lata_Mangeshkar146.png")

image_fromarray = Image.fromarray(image, 'RGB')
resize_image = image_fromarray.resize((128, 128))
expand_input = np.expand_dims(resize_image,axis=0)
input_data = np.array(expand_input)
input_data = input_data/255

pred = loaded_model.predict(input_data)
result = pred.argmax()
print("Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:->",result)
```

```
1/1 [==============================] - 0s 69ms/step
Predicted Emotion{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}:-> 3
```

```
[61] training_set.class_indices
```

```
{'angry': 0, 'happy': 1, 'neutral': 2, 'sad': 3}
```

RESULT AND DISCUSSION

- In conclusion, the emotion classification model developed using a Sequential CNN approach achieved a decent amount of accuracy on training as well as testing data , indicating that it can accurately classify emotions.
- The model's high accuracy is a significant accomplishment, and it demonstrates the effectiveness of using deep learning techniques for emotion classification tasks.
- With further testing on new data, the model can be improved to provide even better results.
- Overall, the project is a success and showcases the potential of deep learning for emotion classification.