Project 2:-

Text Summarization in the Python

I have used following models to test the dataset of BBC News Channel.

1) Bart-large-CNN:- It is a transformer encoder-encoder model with a bidirectional. It is a pre trained model.
2) T5-large:- It is a text to text transformer it can only output class or a span.
3) Pegasus-large:- It is a text to text transformer made by the google.

Bart Large CNN:-

Data:-

```
data1 = """
Gallery unveils interactive tree

A Christmas tree that can receive text messages has been unveiled at
London's Tate Britain art gallery.

The spruce has an antenna which can receive Bluetooth texts sent by
visitors to the Tate. The messages will be "unwrapped" by sculptor Richard
Wentworth, who is responsible for decorating the tree with broken plates
and light bulbs. It is the 17th year that the gallery has invited an
artist to dress their Christmas tree. Artists who have decorated the Tate
tree in previous years include Tracey Emin in 2002.

The plain green Norway spruce is displayed in the gallery's foyer. Its
light bulb adornments are dimmed, ordinary domestic ones joined together
with string. The plates decorating the branches will be auctioned off for
the children's charity ArtWorks. Wentworth worked as an assistant to
sculptor Henry Moore in the late 1960s. His reputation as a sculptor grew
in the 1980s, while he has been one of the most influential teachers
during the last two decades. Wentworth is also known for his photography
of mundane, everyday subjects such as a cigarette packet jammed under the
wonky leg of a table.
"""
```

```
original_summary1 = """
The messages will be "unwrapped" by sculptor Richard Wentworth, who is
responsible for decorating the tree with broken plates and light bulbs.A
Christmas tree that can receive text messages has been unveiled at
London's Tate Britain art gallery.It is the 17th year that the gallery has
invited an artist to dress their Christmas tree.The spruce has an antenna
which can receive Bluetooth texts sent by visitors to the Tate.His
reputation as a sculptor grew in the 1980s, while he has been one of the
most influential teachers during the last two decades.
"""
print(len(data1))
print(len(original_summary1))
```

```
data2= """
Talks held on Gibraltar's future

Two days of talks on the future of Gibraltar begin at Jack Straw's country
residence later on Wednesday.

Officials at the two-day summit at the foreign secretary's official Kent
house, Chevening, will plan a new forum on the Rock's future. In October,
Mr Straw and his Spanish counterpart Miguel Moratinos agreed to establish
a body that would give Gibraltarians a voice in their future. Most
Gibraltarians said in a referendum they wanted to remain British.

Gibraltar's Chief Minister Peter Caruana will represent the British
citizens living on the Rock, while Britain's Europe Director Dominick
Chilcott will represent the UK. Madrid is being represented by Spain's
director general for Europe, Jose Maria Pons. The initiative follows
Spain's socialist government's decision to put its long-standing
sovereignty ambitions on hold. Gibraltarians rejected plans for the Rock's
sovereignty to be shared between Britain and Spain in a referendum
organised by Gibraltar government.
 """

original_summary2 = """
Gibraltarians rejected plans for the Rock's sovereignty to be shared
between Britain and Spain in a referendum organised by Gibraltar
government.Most Gibraltarians said in a referendum they wanted to remain
British.In October, Mr Straw and his Spanish counterpart Miguel Moratinos
agreed to establish a body that would give Gibraltarians a voice in their
future.Officials at the two-day summit at the foreign secretary's official
Kent house, Chevening, will plan a new forum on the Rock's future.
```

```
"""
print(len(data2))
print(len(original_summary2))
```

```
data3= """
Japan narrowly escapes recession

Japan's economy teetered on the brink of a technical recession in the
three months to September, figures show.

Revised figures indicated growth of just 0.1% - and a similar-sized
contraction in the previous quarter. On an annual basis, the data suggests
annual growth of just 0.2%, suggesting a much more hesitant recovery than
had previously been thought. A common technical definition of a recession
is two successive quarters of negative growth.

The government was keen to play down the worrying implications of the
data. "I maintain the view that Japan's economy remains in a minor
adjustment phase in an upward climb, and we will monitor developments
carefully," said economy minister Heizo Takenaka. But in the face of the
strengthening yen making exports less competitive and indications of
weakening economic conditions ahead, observers were less sanguine. "It's
painting a picture of a recovery... much patchier than previously
thought," said Paul Sheard, economist at Lehman Brothers in Tokyo.
Improvements in the job market apparently have yet to feed through to
domestic demand, with private consumption up just 0.2% in the third
quarter.


  """

original_summary3 = """
On an annual basis, the data suggests annual growth of just 0.2%,
suggesting a much more hesitant recovery than had previously been
thought.A common technical definition of a recession is two successive
quarters of negative growth.Revised figures indicated growth of just 0.1%
- and a similar-sized contraction in the previous quarter.Japan's economy
teetered on the brink of a technical recession in the three months to
September, figures show.
"""
print(len(data3))
print(len(original_summary3))
```

Code:-

```
from transformers import BartTokenizer, BartForConditionalGeneration,
T5ForConditionalGeneration, T5Tokenizer, PegasusForConditionalGeneration,
PegasusTokenizer
from rouge import Rouge
```

```

class Agent1_BartLargeCNN:
    def __init__(self, model_name="facebook/bart-large-cnn"):
        self.tokenizer = BartTokenizer.from_pretrained(model_name)
        self.agent1_model =
BartForConditionalGeneration.from_pretrained(model_name)

    def summarize_model(self, data):
        agent1_input_ids = self.tokenizer.encode(data,
return_tensors="pt", max_length=1024, truncation=True)
        agent1_summary_ids = self.agent1_model.generate(agent1_input_ids,
max_length=570, min_length=30, num_beams=2, early_stopping=True)
        agent1_summarized_text =
self.tokenizer.decode(agent1_summary_ids[0], skip_special_tokens=True)
        return agent1_summarized_text

    def calculate_rouge_score(self, predicted, original_summary):
        rouge = Rouge()
        agent1_scores = rouge.get_scores(predicted, original_summary)

        return agent1_scores

agent1_bart = Agent1_BartLargeCNN()

```

Output:-

```
agent1_predicted_summary1 = agent1_bart.summarize_model(data1)


rouge_prediction1 =
agent1_bart.calculate_rouge_score(agent1_predicted_summary1,
original_summary1)

print("Original Data")
print(data1)
print("\n Original Summary:")
print(original_summary1)
print("\n Agent1 Bart Generated Summary:")
print(agent1_predicted_summary1)
print("\n Rouge 1 score prediction for the Agent1 Facebook Bart-Large-
CNN:")
print(rouge_prediction1)
```

Rouge 1 score prediction for the Agent1 Facebook Bart-Large-CNN:

[{'rouge-1': {'r': 0.2361111111111111, 'p': 0.8947368421052632, 'f': 0.3736263703224249}, 'rouge-2': {'r': 0.16129032258064516, 'p': 0.7894736842105263, 'f': 0.2678571400398597}, 'rouge-l': {'r': 0.2361111111111111, 'p': 0.8947368421052632, 'f': 0.3736263703224249}}]

```
agent1_predicted_summary2 = agent1_bart.summarize_model(data2)

rouge_prediction2 =
agent1_bart.calculate_rouge_score(agent1_predicted_summary2,
original_summary2)

print("Original Data")
print(data2)
print("\n Original Summary:")
print(original_summary2)
print("\n Agent1 Bart Generated Summary:")
print(agent1_predicted_summary2)
print("\n Rouge 2 score prediction for the Agent1 Facebook Bart-Large-
CNN:")
print(rouge_prediction2)
```

Rouge 2 score prediction for the Agent1 Facebook Bart-Large-CNN:

[{'rouge-1': {'r': 0.3442622950819672, 'p': 0.75, 'f': 0.47191010804696387}, 'rouge-2': {'r': 0.24324324324324326, 'p': 0.6, 'f': 0.3461538420488166}, 'rouge-l': {'r': 0.32786885245901637, 'p': 0.7142857142857143, 'f': 0.44943819793460042}}]

```
agent1_predicted_summary3 = agent1_bart.summarize_model(data3)

rouge_prediction3 =
agent1_bart.calculate_rouge_score(agent1_predicted_summary3,
original_summary3)

print("Original Data")
print(data3)
print("\n Original Summary:")
print(original_summary3)
print("\n Agent1 Bart Generated Summary:")
print(agent1_predicted_summary3)
print("\n Rouge 3 score prediction for the Agent1 Facebook Bart-Large-
CNN:")
print(rouge_prediction3)
```

Rouge 3 score prediction for the Agent1 Facebook Bart-Large-CNN:

[{'rouge-1': {'r': 0.4727272727272727, 'p': 1.0, 'f': 0.6419753042828837}, 'rouge-2': {'r': 0.4264705882352941, 'p': 0.9666666666666667, 'f': 0.5918367304456477}, 'rouge-l': {'r': 0.4727272727272727, 'p': 1.0, 'f': 0.6419753042828837}}]

T5-large:-

```
class Agent2_T5:
    def __init__(self, model_name="t5-large"):
        self.tokenizer = T5Tokenizer.from_pretrained(model_name)
        self.agent2_model =
T5ForConditionalGeneration.from_pretrained(model_name)

    def summarize_model(self, text):
        agent2_input_ids = self.tokenizer.encode(text,
return_tensors="pt", max_length=1024, truncation=True)
        agent2_summary_ids = self.agent2_model.generate(agent2_input_ids,
max_length=570, min_length=30, do_sample=False)
        agent2_summarized_text =
self.tokenizer.decode(agent2_summary_ids[0], skip_special_tokens=True)
        return agent2_summarized_text
    def evaluate_rouge_score(self, predicted, original_summary):
        rouge = Rouge()
        agent1_scores = rouge.get_scores(predicted, original_summary)
        return agent1_scores

agent2_t5 = Agent2_T5()
```

Output:-

```
agent2_predicted_summary1 = agent2_t5.summarize_model(data1)

rouge_prediction1 =
agent2_t5.evaluate_rouge_score(agent2_predicted_summary1,
original_summary1)

print("Original Data")
print(data1)
print("\n Original Summary:")
print(original_summary1)
print("\n Agent2 T5 Generated Summary:")
print(agent2_predicted_summary1)
print("\n Rouge 1 score prediction for the Agent2 T5 Model:")
print(rouge_prediction1)
```

[{'rouge-1': {'r': 0.2638888888888889, 'p': 0.9047619047619048, 'f': 0.40860214704127645}, 'rouge-2': {'r': 0.1827956989247312, 'p': 0.7083333333333334, 'f': 0.29059828733727816}, 'rouge-l': {'r': 0.2638888888888889, 'p': 0.9047619047619048, 'f': 0.40860214704127645}}]

```
agent2_predicted_summary2 = agent2_t5.summarize_model(data2)

rouge_prediction2 =
agent2_t5.evaluate_rouge_score(agent2_predicted_summary2,
original_summary2)

print("Original Data")
print(data2)
print("\n Original Summary:")
print(original_summary2)
print("\n Agent2 T5 Generated Summary:")
print(agent2_predicted_summary2)
print("\n Rouge 2 score prediction for the Agent2 T5 Model:")
print(rouge_prediction2)
```

Rouge 2 score prediction for the Agent2 T5 Model:

[{'rouge-1': {'r': 0.3442622950819672, 'p': 0.525, 'f': 0.4158415793745711}, 'rouge-2': {'r': 0.1891891891891892, 'p': 0.2978723404255319, 'f': 0.23140495392664445}, 'rouge-l': {'r': 0.3442622950819672, 'p': 0.525, 'f': 0.4158415793745711}}]

```
agent2_predicted_summary3 = agent2_t5.summarize_model(data3)

rouge_prediction3 =
agent2_t5.evaluate_rouge_score(agent2_predicted_summary3,
original_summary3)

print("Original Data")
print(data3)
print("\n Original Summary:")
print(original_summary3)
print("\n Agent2 T5 Generated Summary:")
print(agent2_predicted_summary3)
print("\n Rouge 3 score prediction for the Agent2 T5 Model:")
print(rouge_prediction3)
```

Rouge 3 score prediction for the Agent2 T5 Model:

[{'rouge-1': {'r': 0.10909090909090909, 'p': 0.46153846153846156, 'f': 0.1764705851427336}, 'rouge-2': {'r': 0.014705882352941176, 'p': 0.07142857142857142, 'f': 0.024390241070791525}, 'rouge-l': {'r': 0.10909090909090909, 'p': 0.46153846153846156, 'f': 0.1764705851427336}}]

Pegasus Large

```
class Agent3_Pegasus_large:
    def __init__(self, model_name="google/pegasus-large"):
        self.tokenizer = PegasusTokenizer.from_pretrained(model_name)
        self.model =
PegasusForConditionalGeneration.from_pretrained(model_name)

    def summarize_text(self, text):
        input_ids = self.tokenizer.encode(text, return_tensors="pt",
max_length=1024, truncation=True)
        summary_ids = self.model.generate(input_ids, max_length=150,
min_length=30, num_beams=2, early_stopping=True)
        summarized_text = self.tokenizer.decode(summary_ids[0],
skip_special_tokens=True)
        return summarized_text

    def calculate_rouge_score(self, predicted, original_summary):
        rouge = Rouge()
        agent1_scores = rouge.get_scores(predicted, original_summary)
        return agent1_scores

agent3_pegasus_large = Agent3_Pegasus_large()
```

Output:-

```
agent3_predicted_summary1 = agent3_pegasus_large.summarize_text(data1)

rouge_prediction1 =
agent3_pegasus_large.calculate_rouge_score(agent3_predicted_summary1,
original_summary1)

print("Original Data")
print(data1)
print("\n Original Summary:")
print(original_summary1)
print("\n Agent3 Pegasus Generated Summary:")
print(agent3_predicted_summary1)
print("\n Rouge 1 score prediction for the Agent3 Pegasus:")
print(rouge_prediction1)
```

Rouge 1 score prediction for the Agent3 Pegasus:

```
[{'rouge-1': {'r': 0.5138888888888888, 'p': 0.925, 'f': 0.660714281122449},
'rouge-2': {'r': 0.3978494623655914, 'p': 0.8809523809523809, 'f':
0.5481481438617284}, 'rouge-l': {'r': 0.5138888888888888, 'p': 0.925, 'f':
0.660714281122449}}]
```

```
agent3_predicted_summary2 = agent3_pegasus_large.summarize_text(data2)

rouge_prediction2 =
agent3_pegasus_large.calculate_rouge_score(agent3_predicted_summary2,
original_summary2)

print("Original Data")
print(data2)
print("\n Original Summary:")
print(original_summary2)
print("\n Agent3 Pegasus Generated Summary:")
print(agent3_predicted_summary2)
print("\n Rouge 2 score prediction for the Agent3 Pegasus:")
print(rouge_prediction2)
```

```
Rouge 2 score prediction for the Agent3 Pegasus:

[{'rouge-1': {'r': 0.39344262295081966, 'p': 0.6, 'f': 0.4752475199686305},
'rouge-2': {'r': 0.28378378378378377, 'p': 0.5, 'f': 0.3620689608977408},
'rouge-l': {'r': 0.3770491803278688, 'p': 0.575, 'f': 0.4554455397706107}}]
```

Comparison:-

I have tested the data on the different types of data. Due to that all the model performance depends on the which has got more training on particular data.

All model does not perform well on the political data but all of them perform well on the Entertainment data.

Bart Large CNN model performed well on the each and every data. T5 perform very well on the Entertainment data. But for the other data it needs to be tuned properly to achieved the requires accuracy.

Peagsus of the was consistently performed average recall with but good precision.

The accuracy won't be extremely high because the comparison summary was made by the human and this summary is made by the AI.

How to run:-

```
!pip install transformers rouge tensorflow torch
```

To run the code write

```
Python VaibhavParikh_Project2.py
```