Q1) Calculation of Precision, Recall and Threshold :

Precision : precision can be defined as the ratio of the number of correctly identified instances(TP-True Positives) among all the retrieved positive instances.

It is calculated as :

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall : Recall tells us the proportion of actual positives that have been identified correctly**. When the recall =1, we can say that the model has produced no false Negative.**

It is calculated as :

$$\text{Precision} = \frac{TP}{TP + \text{FN}}$$

F1-Measure : F-measure is a single measure that can be used to represent a trade-off between Precision and Recall .While **F1-Measure is a simple harmonic mean between precision and recall**.

1. The imbalance nature of the dataset is one of the major problems.
2. In such skewed datasets the precision is usually more but the recall value is less. So F-measure can be used to provide a balance value that can be used for model selection.
3. The value for F1-Measure is calculated as follows :

$$\text{F1} - \text{Measure} = \frac{2 * precision * Recall}{precision + Recall}$$

Where, β=1 => Harmonic mean of Precision and Recall,

Example : In a dataset where we need to classify whether a given individual is a terrorist or not ,the no of terrorist will actually be less as compared to a normal individuals .In such a skewed dataset F measure can be a better metric .

Threshold	TP	FN	FP	TN	Correct	Incorrect	Test Set	Precision	Recall	F1 Measure
1	20	80	2	98	100	100	200	0.91	0.2	0.33
5	50	50	5	95	100	100	200	0.91	0.5	0.65
10	60	40	10	90	100	100	200	0.86	0.6	0.71
15	80	20	20	80	100	100	200	0.8	0.8	0.8
20	**88**	**12**	**30**	**70**	**100**	**100**	**200**	**0.75**	**0.88**	**0.81**
25	90	10	40	60	100	100	200	0.69	0.9	0.78
30	95	5	50	50	100	100	200	0.66	0.95	0.78
35	96	4	60	40	100	100	200	0.62	0.96	0.75
40	97	3	70	30	100	100	200	0.58	0.97	0.73
50	98	2	80	20	100	100	200	0.55	0.98	0.71

Observations :

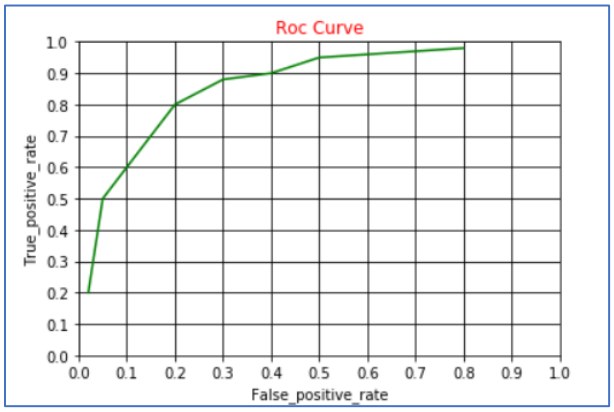
1. For the above calculated data it can be said that the **Threshold of 20 with highest F1-measure of 0.81 is the best threshold point** because :
2. If we select values for threshold which is biased towards either the precision or recall we will have a biased output and no proper results .
3. The F1- Measure when used to select the threshold, both precision and recall values are taken into consideration which will indirectly provide the balance In the skew dataset.

Q2) **ROC** – Curve : Yes, The ROC- Curve can be plotted using the True Positive Rate and False Positive Rate.

True Positive Rate : As mentioned in Question 1 , The True Positive Rate is nothing but the Recall value.

False Positive Rate : False positive rate is the ratio of results that are incorrectly identified as positive to the total number of negatively classified results. Its calculated as : **FP /(TN+FP)**

Threshold	TP	FN	FP	TN	Correct	Incorrect	Test Set	Precision	Recall(TPR)	FPR
1	20	80	2	98	100	100	200	0.91	**0.2**	**0.02**
5	50	50	5	95	100	100	200	0.91	**0.5**	**0.05**
10	60	40	10	90	100	100	200	0.86	**0.6**	**0.1**
15	80	20	20	80	100	100	200	0.8	**0.8**	**0.2**
20	**88**	**12**	**30**	**70**	**100**	**100**	**200**	**0.75**	**0.88**	**0.3**
25	90	10	40	60	100	100	200	0.69	**0.9**	**0.4**
30	95	5	50	50	100	100	200	0.66	**0.95**	**0.5**
35	96	4	60	40	100	100	200	0.62	**0.96**	**0.6**
40	97	3	70	30	100	100	200	0.58	**0.97**	**0.7**
50	98	2	80	20	100	100	200	0.55	**0.98**	**0.8**



ROC Curve

```
import matplotlib.pyplot as plt
fig,ax = plt.subplots()
tpr = [0.2,0.5,0.6,0.8,0.88,0.9,0.95,0.96,0.97,0.98]
fpr= [0.02,0.05,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8]
plt.xlabel('False_positive_rate')
plt.ylabel('True_positive_rate')
ax.plot(fpr,tpr,color="green")
ax.set_title("Roc Curve")
ax.title.set_color('red')
ticks = [0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]
ax.get_xaxis().set_major_formatter(ticker.ScalarFormatter())
ax.get_yaxis().set_major_formatter(ticker.ScalarFormatter())
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.grid(color='black')
plt.axis([0.0,1,0.0,1])
plt.show()
```

Code Snippet

Observation :

The ROC -curve is a measure to evaluate the accuracy of the test, where the area under the ROC curve tells how well a attribute can differentiate between the positively and negatively selected instances.

1. From the above graph plotted for ROC – curve, The TPR = 0.88 and FPR = 0.3 can be identified as a point with threshold of 20, which is the best F1-Measure(i.e. it's the leftmost-top point for our tweets) , it's the best measure for our selected Tweets.
2. The ROC -curve helps us in selecting the model which best suffices over needs based on our requirements of higher precision or Recall or a model which maintains a balance between the precision and recall.
3. The more is the curve towards top-left corner the higher is the accuracy.

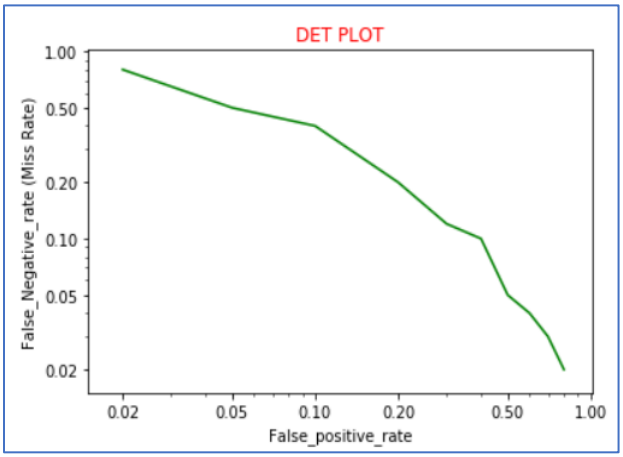
Q3)**DET Curve** : Yes, The DET- Curve can be plotted using the False Negative Rate and False Positive Rate.

False Positive Rate : False positive rate is the ratio of results that are incorrectly identified as positive to the total number of negatively classified results. Its calculated as **FP /(TN+FP)**

False Negative Rate : False Negative Rate is the probablity of getting a false result when actually the required condition is prevalent. It is calculated as :

$$FNR = \frac{FN}{TP + FN}$$

Threshold	TP	FN	FP	TN	Correct	Incorrect	Test Set	Precision	FNR	FPR
1	20	80	2	98	100	100	200	0.91	**0.8**	**0.02**
5	50	50	5	95	100	100	200	0.91	**0.5**	**0.05**
10	60	40	10	90	100	100	200	0.86	**0.4**	**0.1**
15	80	20	20	80	100	100	200	0.8	**0.2**	**0.2**
20	88	12	30	70	100	100	200	0.75	**0.12**	**0.3**
25	90	10	40	60	100	100	200	0.69	**0.1**	**0.4**
30	95	5	50	50	100	100	200	0.66	**0.05**	**0.5**
35	96	4	60	40	100	100	200	0.62	**0.04**	**0.6**
40	97	3	70	30	100	100	200	0.58	**0.03**	**0.7**
50	98	2	80	20	100	100	200	0.55	**0.02**	**0.8**



DET Curve

```
import matplotlib.pyplot as plt
fig,ax = plt.subplots()
fpr=[0.02,0.05,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8]
fnr=[0.8,0.5,0.4,0.2,0.12,0.1,0.05,0.04,0.03,0.02]
plt.xlabel('False_positive_rate')
plt.ylabel('False_Negative_rate (Miss Rate)')
plt.yscale('log')
plt.xscale('log')
ax.plot(fpr,fnr,color="green")
ax.set_title("DET PLOT")
ax.title.set_color('red')
ticks = [0.02,0.05,0.1,0.2,0.5,1]
ax.get_xaxis().set_major_formatter(ticker.ScalarFormatter())
ax.get_yaxis().set_major_formatter(ticker.ScalarFormatter())
ax.set_xticks(ticks)
ax.set_yticks(ticks)
#x.grid(color='black')
plt.axis([0.015,1.015,0.015,1.015])
plt.show()
```

Code Snippet

DET Curve (Detection Error Trade-off):

The DET- Curve represents the trade off between the False Positive Rate (FPR) and the Miss Rate (FNR-False Negative Rate).

While plotting the DET Curve the axes are plotted using the logarithmic scale to focus on the Key Details of ROC Curve.

For our data we don't have the best possible model because, the best model is the one that has both a low miss rate as well as a low False Positive Rate, i.e. when the DET Curve is more inclined toward the lower left corner of the plot.

References :

1. https://matplotlib.org/api/_as_gen/matplotlib.pyplot.xticks.html
2. https://www.quora.com/What-is-the-best-example-for-false-negative-false-positive-true-negative-and-true-positive-in-machine-learning
3. https://stackoverflow.com/questions/29630737/matplotlib-line-plot-of-x-values-against-y
4. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall