**PRACTICAL NO 4**

**AIM :** Evaluation Metrics for IR Systems

* Calculate precision, recall, and F-measure for a given set of retrieval results.
* Use an evaluation toolkit to measure average precision and other evaluation metrics.

**SOLUTION:**

1) Calculate precision, recall, and F-measure for a given set of retrieval results.

**INPUT:**

def calculate\_metrics(retrieved\_set, relevant\_set):

tp = len(retrieved\_set & relevant\_set) # True Positives

fp = len(retrieved\_set - relevant\_set) # False Positives

fn = len(relevant\_set - retrieved\_set) # False Negatives

print(f"True Positive: {tp}\nFalse Positive: {fp}\nFalse Negative: {fn}\n")

precision = tp / (tp + fp) if (tp + fp) > 0 else 0

recall = tp / (tp + fn) if (tp + fn) > 0 else 0

f\_measure = 2 \* precision \* recall / (precision + recall) if (precision + recall) > 0 else 0

return precision, recall, f\_measure

# Example input

retrieved\_set = {"doc1", "doc2", "doc3"} # Predicted set

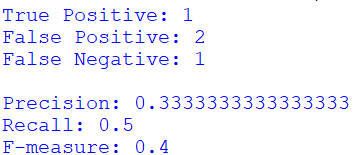
relevant\_set = {"doc1", "doc4"} # Relevant set

# Calculate and display metrics

precision, recall, f\_measure = calculate\_metrics(retrieved\_set, relevant\_set)

print(f"Precision: {precision}\nRecall: {recall}\nF-measure: {f\_measure}")

**OUTPUT:**



2) Use an evaluation toolkit to measure average precision and other evaluation metrics.

**INPUT:**

from sklearn.metrics import average\_precision\_score

y\_true = [0, 1, 1, 0, 1, 1] #Binary Prediction

y\_scores = [0.1, 0.4, 0.35, 0.8, 0.65, 0.9] #Model's estimation score

average\_precision = average\_precision\_score(y\_true, y\_scores)

print(f'Average precision-recall score: {average\_precision}')

OUTPUT:

