Project Description:

Perform sentiment analysis on three types of text:

1. Movie reviews
2. Product reviews
3. Tweets

Two systems are implemented:

1. A classical machine learning sentiment classifier
2. A transformer-based sentiment classifier using BERT

The goal is to compare traditional feature-based models with modern transformer models using the same datasets and evaluation metrics.

In [25]:
```
# !pip install scikit-learn pandas nltk transformers datasets (Install it initia
```

In [26]:
```
# Install dependencies
# !pip install -q datasets transformers scikit-learn pandas nltk (Install it ini
```

In [27]:
```
# Import Libraries
import pandas as pd
import numpy as np
from datasets import load_dataset
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_sc
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
from transformers import pipeline
import random
```

In [28]:
```
# NLTK downloads
nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[28]: True

Datasets Used

Three publicly available datasets were used in this project:

- IMDB movie reviews for sentiment classification
- Amazon Polarity dataset for product reviews
- Sentiment140 dataset for Twitter sentiment analysis

Each dataset contains positive and negative sentiment labels. A balanced subset of 100 samples was selected from each dataset to ensure fair and efficient experimentation.

In [29]:
```
# 1. Dataset Load
# Movie reviews (IMDB)
imdb = load_dataset("imdb")
df_movie = pd.DataFrame(imdb['train'].shuffle(seed=42).select(range(100)))[["tex
# Mapping labels 0->negative, 1->positive
df_movie['label'] = df_movie['label'].map({0: 'negative', 1: 'positive'})
```

In [31]:
```
# Product reviews (Amazon Polarity) - Amazon Reviews
amazon = load_dataset("amazon_polarity")
df_product = pd.DataFrame(amazon['train'].shuffle(seed=42).select(range(100)))[[
df_product.rename(columns={"content":"text"}, inplace=True)
df_product['label'] = df_product['label'].map({0: 'negative', 1: 'positive'})
```

In [32]:
```
# Tweets (Sentiment140 or Twitter US Airline Sentiment): Can use any of the data

import pandas as pd
import kagglehub
import os

# Loading the Sentiment140 dataset
# Dataset Source Link: https://www.kaggle.com/datasets/kazanova/sentiment140
dataset_path = kagglehub.dataset_download("kazanova/sentiment140")
print("Dataset downloaded to:", dataset_path)

# The CSV file inside is usually named "training.1600000.processed.noemoticon.cs
csv_file = os.path.join(dataset_path, "training.1600000.processed.noemoticon.csv

# Loading the CSV File into pandas
df_tweets = pd.read_csv(csv_file, encoding='latin-1', names=["target", "id", "da

# Filtering only positive (4) and negative (0) tweets with labels
df_tweets = df_tweets[df_tweets['target'].isin([0, 4])]

# 100 tweets sample
neg = df_tweets[df_tweets["target"] == 0].sample(50, random_state=42)
pos = df_tweets[df_tweets["target"] == 4].sample(50, random_state=42)

df_tweets = pd.concat([neg, pos]).sample(frac=1, random_state=42)
df_tweets["label"] = df_tweets["target"].map({0: "negative", 4: "positive"})
df_tweets = df_tweets[["text", "label"]]

print("Tweets dataset loaded. Sample:")
print(df_tweets.head())
```

```
Using Colab cache for faster access to the 'sentiment140' dataset.
Dataset downloaded to: /kaggle/input/sentiment140
Tweets dataset loaded. Sample:
                                                      text     label
1534779                  @saturnboy good job!  So pretty   positive
1388988  @hillsongunited can't wait to worship with you...  positive
1403776  Of course, as always, Jerry wants some tweaks ...  positive
722634   one month ago i was the most happiest person.....  negative
325297   once the concert stories start rollin in I mig...  negative
```

In [33]:
```python
datasets = {
    "Movie": df_movie,
    "Product": df_product,
    "Tweets": df_tweets
}
```

Text Preprocessing

Text preprocessing is applied only for the classical machine learning models.

Steps include:

- Converting text to lowercase for consistency
- Tokenizing text into individual words
- Removing stopwords to reduce noise
- Applying lemmatization to normalize word forms

These steps help improve model performance by reducing irrelevant variations.

In [34]:
```python
# 2. Preprocessing (for Classical ML Sentiment Classifier)
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def preprocess(text):
    text = text.lower()
    tokens = word_tokenize(text)
    tokens = [lemmatizer.lemmatize(tok) for tok in tokens if tok.isalpha() and t
    return " ".join(tokens)

for name, df in datasets.items():
    df['clean_text'] = df['text'].astype(str).apply(preprocess)
```

Classical Machine Learning Models

Logistic Regression and Naive Bayes are used as classical sentiment classifiers. These models rely on manually engineered features such as TF-IDF vectors.

Both models are trained on the same training set and evaluated on a held-out test set.

Note: Feature Engineering

TF-IDF (Term Frequency–Inverse Document Frequency) is used to convert text into numerical feature vectors.

TF-IDF assigns higher importance to words that are meaningful within a document while reducing the influence of commonly occurring words. The vocabulary size is printed to

understand the dimensionality of the feature space.

In [35]:
```
# 3. Classical ML Sentiment Classifier: Vectorize + Train + Evaluate (System A)
results = {}
test_sets = {}

for name, df in datasets.items():
    if df.empty:
        continue
    print(f"\n===== {name} Dataset (Classical ML) =====")
    X = df['clean_text']
    y = df['label']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran
    test_sets[name] = (X_test, y_test)

    vectorizer = TfidfVectorizer()  # or CountVectorizer() for Bag-of-Words
    X_train_vec = vectorizer.fit_transform(X_train)
    X_test_vec = vectorizer.transform(X_test)

    print("Vocabulary size:", len(vectorizer.vocabulary_))

    # Logistic Regression Classifier
    clf = LogisticRegression(max_iter=1000)
    clf.fit(X_train_vec, y_train)
    y_pred = clf.predict(X_test_vec)

    acc = accuracy_score(y_test, y_pred)
    prec = precision_score(y_test, y_pred, pos_label='positive')
    rec = recall_score(y_test, y_pred, pos_label='positive')
    f1 = f1_score(y_test, y_pred, pos_label='positive')
    print("\nLogistic Regression Classifier Results:")
    print("Accuracy:", acc)
    print("Precision:", prec)
    print("Recall:", rec)
    print("F1-score:", f1)
    print("\nClassification Report:\n", classification_report(y_test, y_pred))

    # Naive Bayes Classifier
    nb = MultinomialNB()
    nb.fit(X_train_vec, y_train)
    nb_pred = nb.predict(X_test_vec)

    nb_acc = accuracy_score(y_test, nb_pred)
    nb_prec = precision_score(y_test, nb_pred, pos_label='positive')
    nb_rec = recall_score(y_test, nb_pred, pos_label='positive')
    nb_f1 = f1_score(y_test, nb_pred, pos_label='positive')

    print("\nNaive Bayes Classifier Results:")
    print("Accuracy:", nb_acc)
    print("Precision:", nb_prec)
    print("Recall:", nb_rec)
    print("F1-score:", nb_f1)

    # Results
    results[name + "_LR"] = {
    "model": "Logistic Regression",
    "accuracy": acc,
    "precision": prec,
```

```
    "recall": rec,
    "f1": f1}

    results[name + "_NB"] = {
    "model": "Naive Bayes",
    "accuracy": nb_acc,
    "precision": nb_prec,
    "recall": nb_rec,
    "f1": nb_f1}

    print("\nSome test samples with predictions:")
    for i, text in enumerate(X_test.iloc[:5]):
        print("Text:", text)
        print("True label:", y_test.iloc[i], "Pred:", y_pred[i])
        print("----")

    print("\nExample Correct vs Incorrect Predictions:")

    correct_example = None
    incorrect_example = None

    for i in range(len(y_test)):
        if y_test.iloc[i] == y_pred[i] and correct_example is None:
            correct_example = i
        if y_test.iloc[i] != y_pred[i] and incorrect_example is None:
            incorrect_example = i
        if correct_example is not None and incorrect_example is not None:
            break

    if correct_example is not None:
        print("\nCorrect Prediction Example:")
        print("Text:", X_test.iloc[correct_example][:200])
        print("True Label:", y_test.iloc[correct_example])
        print("Predicted Label:", y_pred[correct_example])

    if incorrect_example is not None:
        print("\nIncorrect Prediction Example:")
        print("Text:", X_test.iloc[incorrect_example][:200])
        print("True Label:", y_test.iloc[incorrect_example])
        print("Predicted Label:", y_pred[incorrect_example])
```

```
===== Movie Dataset (Classical ML) =====
Vocabulary size: 3796

Logistic Regression Classifier Results:
Accuracy: 0.65
Precision: 1.0
Recall: 0.2222222222222222
F1-score: 0.36363636363636365

Classification Report:
              precision    recall  f1-score   support

    negative       0.61      1.00      0.76        11
    positive       1.00      0.22      0.36         9

    accuracy                           0.65        20
   macro avg       0.81      0.61      0.56        20
weighted avg       0.79      0.65      0.58        20


Naive Bayes Classifier Results:
Accuracy: 0.6
Precision: 1.0
Recall: 0.1111111111111111
F1-score: 0.2

Some test samples with predictions:
Text: gu van sant made excellent film truly br br however ca help feel cerebral e
dge tom robbins book even cowgirl get blue lost translation big screen alone tom
robbins gu van sant incredible visionary tower talent ultimately though one work
br br character well developed plot content come alive imagination much powerful
reading book like taken away different time place sometimes think worst best add
overall book neatly unfolds according author precision movie however leave one le
ss imagination emotion detracting overall experience believe happened br br sugge
st reading book
True label: positive Pred: negative
----
Text: sister said movie gon na good second thought watched actually funny basical
ly movie made weird girl go small town one like want go get reading aunt go easy
movie come across hilarious humor witch book mentally challenged uncle dog unders
tand meaning word freak anyways hope run right try find really old movie hope lik
e total give totally joking ill give hope understand laugh scream may br br love
truly dakota email hot wan na
True label: positive Pred: negative
----
Text: unimpressed cinderella jungle book possibly worse title first like animatio
n worse scene liked character namely thunderbolt patch character like cruella med
iocre cruella truly villainous original lost quality sequel said nothing write ho
me animation kind ugly also artist companion lars joke honest roger seemed quit s
moking overnight voice talent good though especially barry bostwick thunderbolt e
xception jodi benson accent ruined good moment whole plot seemed bloated highly s
uggestive extended tv episode hugely disappointing sequel memorable disney movie
along jungle book sorry give cup tea bethany cox
True label: negative Pred: negative
----
Text: opinion pretty good celebrity skit show enjoyed seeing greg kinnear host ma
ny reason said even though hal spark okay host sometimes wish greg kinnear left a
sk seems nobody stay tv show throughout entire run anymore still enjoyed seeing v
arious host people spoofing celebrity ask pretty darn funny wrap must say kind mi
ss show conclusion highly recommend show sketch show fan really enjoy
```

```
True label: positive Pred: negative
----
Text: earth colin firth pointless film really strapped cash br br film clear want
grief exotic place ghost vehicle mr darcy muddled muddy br br seems sort idea ita
ly must good italian something offer language end girl want go back br br pointle
ss episode beach church busy road anybody care simply br br also yank woman film
clear job seemed make vapid inappropriate maudlin comment girl supposed paedophil
lia br br pretty dreadful mess gave rather charm utterly ghastly film
True label: negative Pred: negative
----

Example Correct vs Incorrect Predictions:

Correct Prediction Example:
Text: unimpressed cinderella jungle book possibly worse title first like animatio
n worse scene liked character namely thunderbolt patch character like cruella med
iocre cruella truly villainous original lost
True Label: negative
Predicted Label: negative

Incorrect Prediction Example:
Text: gu van sant made excellent film truly br br however ca help feel cerebral e
dge tom robbins book even cowgirl get blue lost translation big screen alone tom
robbins gu van sant incredible visionary tow
True Label: positive
Predicted Label: negative

===== Product Dataset (Classical ML) =====
Vocabulary size: 1524

Logistic Regression Classifier Results:
Accuracy: 0.55
Precision: 0.6
Recall: 0.3
F1-score: 0.4

Classification Report:
              precision    recall  f1-score   support

    negative       0.53      0.80      0.64        10
    positive       0.60      0.30      0.40        10

    accuracy                           0.55        20
   macro avg       0.57      0.55      0.52        20
weighted avg       0.57      0.55      0.52        20


Naive Bayes Classifier Results:
Accuracy: 0.5
Precision: 0.5
Recall: 0.4
F1-score: 0.4444444444444444

Some test samples with predictions:
Text: unit work well nice long cord good value ear pad bit small
True label: positive Pred: negative
----
Text: son soes like variable nipple tried one take bottle well problem hard find
store buy extra found
True label: positive Pred: negative
```

```
----
Text: bought iron two year ago forever leaving water even big mistake spit water
well heating element work properly iron made take abuse get another however buy a
nother like love titanium finish soleplate ca tell many time used fusible fabric
never stuck much better surface used wish could get iron repaired everything else
work fine
True label: positive Pred: negative
----
Text: ordered book christmas gift least friend family avid reader least book week
usually forget book shortly read book hard forget week later still recall whole c
hapter touched life much want share many people possible highly recommend year la
ter one book gave friend died unexpectantly talked book many time know room marve
l helped deal loss know touched life book
True label: positive Pred: positive
----
Text: given gift soon realized product flawed player connected pc battery dy corr
upt internal memory player restart replacing battery player bios initiate however
internal memory check complete message battery low appear shuts internal memory p
layer must contacted digitalway regarding said player purchase date less day old
receipt may return place purchase player purchase date day less day old receipt m
ay return digitalway key receipt also believe really backing product would recomm
end looking comparable player especially warranty
True label: negative Pred: negative
----

Example Correct vs Incorrect Predictions:

Correct Prediction Example:
Text: ordered book christmas gift least friend family avid reader least book week
usually forget book shortly read book hard forget week later still recall whole c
hapter touched life much want share many pe
True Label: positive
Predicted Label: positive

Incorrect Prediction Example:
Text: unit work well nice long cord good value ear pad bit small
True Label: positive
Predicted Label: negative

===== Tweets Dataset (Classical ML) =====
Vocabulary size: 451

Logistic Regression Classifier Results:
Accuracy: 0.6
Precision: 0.5833333333333334
Recall: 0.7
F1-score: 0.6363636363636364

Classification Report:
              precision    recall  f1-score   support

    negative       0.62      0.50      0.56        10
    positive       0.58      0.70      0.64        10

    accuracy                           0.60        20
   macro avg       0.60      0.60      0.60        20
weighted avg       0.60      0.60      0.60        20


Naive Bayes Classifier Results:
```

```
Accuracy: 0.7
Precision: 0.75
Recall: 0.6
F1-score: 0.6666666666666666

Some test samples with predictions:
Text: allright good night amp thank much hug
True label: positive Pred: negative
----
Text: mad late good luck
True label: positive Pred: positive
----
Text: aquaahh ilove everything old school
True label: positive Pred: positive
----
Text: hillsongunited ca wait worship guy tonight much fun
True label: positive Pred: negative
----
Text: anniegxxx ca think way least try lol
True label: negative Pred: positive
----

Example Correct vs Incorrect Predictions:

Correct Prediction Example:
Text: mad late good luck
True Label: positive
Predicted Label: positive

Incorrect Prediction Example:
Text: allright good night amp thank much hug
True Label: positive
Predicted Label: negative
```

Example Predictions Analysis:

The correctly classified example shows clear sentiment-related keywords, making it easier for the model to predict accurately. The incorrect example highlights cases where sentiment may be subtle, ambiguous, or context-dependent, which classical models often struggle with.

Transformer-Based Sentiment Analysis (BERT)

BERT is a transformer-based model that captures contextual meaning in text.

The pre-trained DistilBERT model fine-tuned on SST-2 is used via the HuggingFace pipeline.

Unlike classical models, BERT does not require manual preprocessing or feature engineering.

The model is evaluated on the same test samples used for classical models to ensure fairness.

In [36]:
```
# 4. BERT-based sentiment (System B: Transformer-Based Sentiment Model) (System
# BERT - Bidirectional Encoder Representations from Transformers.
print("\nTransformer-Based Sentiment Model(BERT)")
```

```
bert_clf = pipeline(
    "sentiment-analysis",
    model="distilbert-base-uncased-finetuned-sst-2-english",
    truncation=True,   # Ensuring texts >512 tokens are truncated
    max_length=512
)
results_bert = {}

for name, df in datasets.items():
    if df.empty:
        continue
    print(f"\n--- {name} dataset (BERT) ---")

    # Using SAME test set as Classical ML
    X_test, y_test = test_sets[name]

    texts = X_test.astype(str).tolist()
    true_labels = y_test.tolist()

    preds = bert_clf(texts)
    bert_labels = [pred['label'].lower() for pred in preds]

    # Evaluation metrics
    acc = accuracy_score(true_labels, bert_labels)
    prec = precision_score(true_labels, bert_labels, pos_label='positive')
    rec = recall_score(true_labels, bert_labels, pos_label='positive')
    f1 = f1_score(true_labels, bert_labels, pos_label='positive')
    print(f"BERT metrics → Accuracy: {acc:.2f}, Precision: {prec:.2f}, Recall: {

    results_bert[name] = {"model": "BERT", "accuracy": acc, "precision": prec, "

    # Sample test predictions
    for t, p in zip(texts[:10], preds[:10]):
        print("Text:", t[:200], "...")  # printing only first 200 chars for read
        print("BERT →", p)
        print("-----")
```

```
Transformer-Based Sentiment Model(BERT)
```

```
Device set to use cpu
```

```
--- Movie dataset (BERT) ---
BERT metrics → Accuracy: 0.85, Precision: 1.00, Recall: 0.67, F1-Score: 0.80
Text: gu van sant made excellent film truly br br however ca help feel cerebral e
dge tom robbins book even cowgirl get blue lost translation big screen alone tom
robbins gu van sant incredible visionary tow ...
BERT → {'label': 'POSITIVE', 'score': 0.9903706908226013}
-----
Text: sister said movie gon na good second thought watched actually funny basical
ly movie made weird girl go small town one like want go get reading aunt go easy
movie come across hilarious humor witch book ...
BERT → {'label': 'NEGATIVE', 'score': 0.8905219435691833}
-----
Text: unimpressed cinderella jungle book possibly worse title first like animatio
n worse scene liked character namely thunderbolt patch character like cruella med
iocre cruella truly villainous original lost ...
BERT → {'label': 'NEGATIVE', 'score': 0.9994425177574158}
-----
Text: opinion pretty good celebrity skit show enjoyed seeing greg kinnear host ma
ny reason said even though hal spark okay host sometimes wish greg kinnear left a
sk seems nobody stay tv show throughout enti ...
BERT → {'label': 'POSITIVE', 'score': 0.9976315498352051}
-----
Text: earth colin firth pointless film really strapped cash br br film clear want
grief exotic place ghost vehicle mr darcy muddled muddy br br seems sort idea ita
ly must good italian something offer langua ...
BERT → {'label': 'NEGATIVE', 'score': 0.9980921149253845}
-----
Text: finished watching movie ridiculously bad really disappointed really sure so
meone would make movie like marginally entertaining feel like people making lot d
isagreement making monday writer charge tues ...
BERT → {'label': 'NEGATIVE', 'score': 0.9989960789680481}
-----
Text: someone staggered incredible visuals hero anxious see film billed along lin
e better also featured actress like ziyi zhang well disappointed count bought dvd
film mistake br br realize martial art film ...
BERT → {'label': 'NEGATIVE', 'score': 0.9925881028175354}
-----
Text: soprano probably widely acclaimed tv series ever naturally expectation roof
yet show surpassed love mafia crime genre film enjoy following compelling story s
et world much hour material give story chan ...
BERT → {'label': 'POSITIVE', 'score': 0.996277391910553}
-----
Text: think favorite part movie one exemplifies sheer pointless stupidity inanity
proceeding come climax film doctor ted nelson unmarried friend sheriff finally co
rnered melting man landing stair electrical ...
BERT → {'label': 'NEGATIVE', 'score': 0.9978529214859009}
-----
Text: film failed explore humanity animal left empty feeling inside spoiler ahead
convinced really compelling reason forego big buyout deal help furry friend where
as babe original bucked trend hit focusing  ...
BERT → {'label': 'NEGATIVE', 'score': 0.9983041286468506}
-----

--- Product dataset (BERT) ---
BERT metrics → Accuracy: 0.65, Precision: 0.71, Recall: 0.50, F1-Score: 0.59
Text: unit work well nice long cord good value ear pad bit small ...
BERT → {'label': 'POSITIVE', 'score': 0.9939022064208984}
-----
Text: son soes like variable nipple tried one take bottle well problem hard find
store buy extra found ...
```

```
BERT → {'label': 'NEGATIVE', 'score': 0.9971779584884644}
-----
Text: bought iron two year ago forever leaving water even big mistake spit water
well heating element work properly iron made take abuse get another however buy a
nother like love titanium finish soleplate c ...
BERT → {'label': 'NEGATIVE', 'score': 0.845703125}
-----
Text: ordered book christmas gift least friend family avid reader least book week
usually forget book shortly read book hard forget week later still recall whole c
hapter touched life much want share many pe ...
BERT → {'label': 'POSITIVE', 'score': 0.9756944179534912}
-----
Text: given gift soon realized product flawed player connected pc battery dy corr
upt internal memory player restart replacing battery player bios initiate however
internal memory check complete message batt ...
BERT → {'label': 'NEGATIVE', 'score': 0.9973987340927124}
-----
Text: battery received apc picture description suggested able return ...
BERT → {'label': 'NEGATIVE', 'score': 0.9395267963409424}
-----
Text: sorry say movie disappointment harrison ford best expected better make conv
incing cowboy indian seemed rather lame movie character well developed beginning
story boring movie plot well done decent par ...
BERT → {'label': 'NEGATIVE', 'score': 0.9080843329429626}
-----
Text: looked planer going buy planer amp first concern amp motor planer fail put
machine work rough cut white oak blade full bore got tired hitachi planer wish sh
oe planer class inch longer longer shoe base ...
BERT → {'label': 'NEGATIVE', 'score': 0.996464729309082}
-----
Text: gruen writes like woman trying sound like man word choice syntax ring true
unfortunately method cost credibility book nothing write home certainly rank anyt
hing erik larson high rating reviewer puzzli ...
BERT → {'label': 'POSITIVE', 'score': 0.633484423160553}
-----
Text: although metal extremely flimsy hold disc securely total waste money ...
BERT → {'label': 'NEGATIVE', 'score': 0.998576283454895}
-----

--- Tweets dataset (BERT) ---
BERT metrics → Accuracy: 0.90, Precision: 0.90, Recall: 0.90, F1-Score: 0.90
Text: allright good night amp thank much hug ...
BERT → {'label': 'POSITIVE', 'score': 0.999849796295166}
-----
Text: mad late good luck ...
BERT → {'label': 'POSITIVE', 'score': 0.9998016953468323}
-----
Text: aquaahh ilove everything old school ...
BERT → {'label': 'NEGATIVE', 'score': 0.9765186905860901}
-----
Text: hillsongunited ca wait worship guy tonight much fun ...
BERT → {'label': 'POSITIVE', 'score': 0.9957802295684814}
-----
Text: anniegxxx ca think way least try lol ...
BERT → {'label': 'NEGATIVE', 'score': 0.9956562519073486}
-----
Text: amytheallen assume check actually win something ...
BERT → {'label': 'NEGATIVE', 'score': 0.9946137070655823}
-----
Text: xpowxbangxboomx dammiitt wish mtv ...
```

```
BERT → {'label': 'NEGATIVE', 'score': 0.9869897961616516}
-----
Text: sillybeggar congrats james sure book going huge success ...
BERT → {'label': 'POSITIVE', 'score': 0.999553382396698}
-----
Text: grounded ...
BERT → {'label': 'POSITIVE', 'score': 0.9650882482528687}
-----
Text: divacandicem candice dont mean bother im really sad u got released ...
BERT → {'label': 'NEGATIVE', 'score': 0.9736489653587341}
-----
```

In [37]:
```python
# Summary Table
summary = pd.DataFrame([
    {
        "Dataset": k.replace("_LR","").replace("_NB",""),
        "Model": v["model"],
        "Accuracy": v["accuracy"],
        "Precision": v["precision"],
        "Recall": v["recall"],
        "F1-score": v["f1"]
    }
    for k, v in results.items()
] + [
    {
        "Dataset": k,
        "Model": "BERT",
        "Accuracy": v["accuracy"],
        "Precision": v["precision"],
        "Recall": v["recall"],
        "F1-score": v["f1"]
    }
    for k, v in results_bert.items()
])
print("\n========== Summary Table ==========")
print(summary)
```

```
========== Summary Table ==========
   Dataset                Model  Accuracy  Precision    Recall  F1-score
0    Movie  Logistic Regression      0.65   1.000000  0.222222  0.363636
1    Movie          Naive Bayes      0.60   1.000000  0.111111  0.200000
2  Product  Logistic Regression      0.55   0.600000  0.300000  0.400000
3  Product          Naive Bayes      0.50   0.500000  0.400000  0.444444
4   Tweets  Logistic Regression      0.60   0.583333  0.700000  0.636364
5   Tweets          Naive Bayes      0.70   0.750000  0.600000  0.666667
6    Movie                 BERT      0.85   1.000000  0.666667  0.800000
7  Product                 BERT      0.65   0.714286  0.500000  0.588235
8   Tweets                 BERT      0.90   0.900000  0.900000  0.900000
```

Evaluation and Comparison Analysis

The results show a clear performance difference between classical machine learning models and the BERT-based transformer model across all three datasets.

Movie Reviews

For movie reviews, both Logistic Regression and Naive Bayes achieve high precision but very low recall. This indicates that while the models correctly predict positive sentiment

when confident, they miss many positive examples. In contrast, BERT significantly improves this balance, achieving higher accuracy and F1-score by effectively capturing contextual information in longer reviews.

Product Reviews

In the product reviews dataset, classical models demonstrate moderate performance and struggle with mixed or subtle sentiment expressions. BERT again outperforms the classical approaches, showing improved accuracy and a higher F1-score, which suggests a stronger ability to understand nuanced opinions.

Tweets

Tweets often contain informal language and limited context, making sentiment classification more challenging. Classical models show inconsistent performance, with Naive Bayes slightly outperforming Logistic Regression. However, BERT clearly achieves the best results, delivering the highest accuracy along with well-balanced precision and recall.

Overall Comparison

Overall, the results confirm that while classical machine learning models are computationally efficient and interpretable, they are limited in handling contextual and informal language. The BERT-based model consistently provides superior sentiment classification performance due to its ability to capture contextual meaning in text.