In [1]:
```
!pip install pandas numpy matplotlib scikit-learn
```

```
Requirement already satisfied: pandas in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (2.2.3)
Requirement already satisfied: numpy in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (2.1.3)
Requirement already satisfied: matplotlib in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (3.10.
0)
Requirement already satisfied: scikit-learn in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (1.
7.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\adithyaa\appdata\roaming\python\python313\site-packages (from
pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (fro
m pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (f
rom pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages
(from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (fro
m matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages
(from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages
(from matplotlib) (1.4.9)
Requirement already satisfied: packaging>=20.0 in c:\users\adithyaa\appdata\roaming\python\python313\site-packages (from matplo
tlib) (25.0)
Requirement already satisfied: pillow>=8 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (from m
atplotlib) (12.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages
(from matplotlib) (3.2.5)
Requirement already satisfied: scipy>=1.8.0 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (fro
m scikit-learn) (1.15.3)
Requirement already satisfied: joblib>=1.2.0 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packages (fr
om scikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\adithyaa\appdata\local\programs\python\python313\lib\site-packa
ges (from scikit-learn) (3.6.0)
Requirement already satisfied: six>=1.5 in c:\users\adithyaa\appdata\roaming\python\python313\site-packages (from python-dateut
il>=2.8.2->pandas) (1.17.0)
```

```
[notice] A new release of pip is available: 25.0.1 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [2]:
```
import os
import re
import string
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB, GaussianNB
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score, f1_score,
    confusion_matrix, ConfusionMatrixDisplay,
    roc_curve, auc, classification_report
)
```

In [3]:
```
df = pd.read_csv("C:\\Users\\Adithyaa\\Downloads\\spam.csv", encoding="latin-1")

# keep only useful columns
df = df[['v1', 'v2']]
df.columns = ['label', 'text']

print("✅ Dataset Loaded Successfully")
print(df.head())
print("\nLabel Counts:")
print(df['label'].value_counts())
```

```
✅ Dataset Loaded Successfully
  label                                               text
0   ham  Go until jurong point, crazy.. Available only ...
1   ham                      Ok lar... Joking wif u oni...
2  spam  Free entry in 2 a wkly comp to win FA Cup fina...
3   ham  U dun say so early hor... U c already then say...
4   ham  Nah I don't think he goes to usf, he lives aro...

Label Counts:
label
ham     4825
spam     747
Name: count, dtype: int64
```

In [4]:
```
df["label"] = df["label"].map({"ham": 0, "spam": 1})

STOPWORDS = {
    "a","an","the","and","or","but","if","while","with","to","from","of","in","on",
    "for","at","by","is","are","was","were","be","been","this","that","it","as",
    "i","you","he","she","we","they","me","my","your","our","their","so","do","does","did"
}

def clean_text(text):
    text = text.lower()
    text = re.sub(r"http\S+|www\S+|https\S+", "", text)      # remove links
    text = re.sub(r"\d+", "", text)                             # remove numbers
    text = text.translate(str.maketrans("", "", string.punctuation))  # punctuation
    text = re.sub(r"\s+", " ", text).strip()                    # remove extra spaces
    words = [w for w in text.split() if w not in STOPWORDS]  # stopwords removal
    return " ".join(words)

df["clean_text"] = df["text"].apply(clean_text)

print("\n✅ Text Preprocessing Done")
print(df.head())
```

```
✅ Text Preprocessing Done
   label                                               text  \
0      0  Go until jurong point, crazy.. Available only ...
1      0                      Ok lar... Joking wif u oni...
2      1  Free entry in 2 a wkly comp to win FA Cup fina...
3      0  U dun say so early hor... U c already then say...
4      0  Nah I don't think he goes to usf, he lives aro...

                                         clean_text
0  go until jurong point crazy available only bug...
1                           ok lar joking wif u oni
2  free entry wkly comp win fa cup final tkts st ...
3                u dun say early hor u c already then say
4   nah dont think goes usf lives around here though
```

In [5]:
```
X = df["clean_text"]
y = df["label"]
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

print("\n✅ Dataset Split Done")
print("Train size:", len(X_train))
print("Test size :", len(X_test))
```

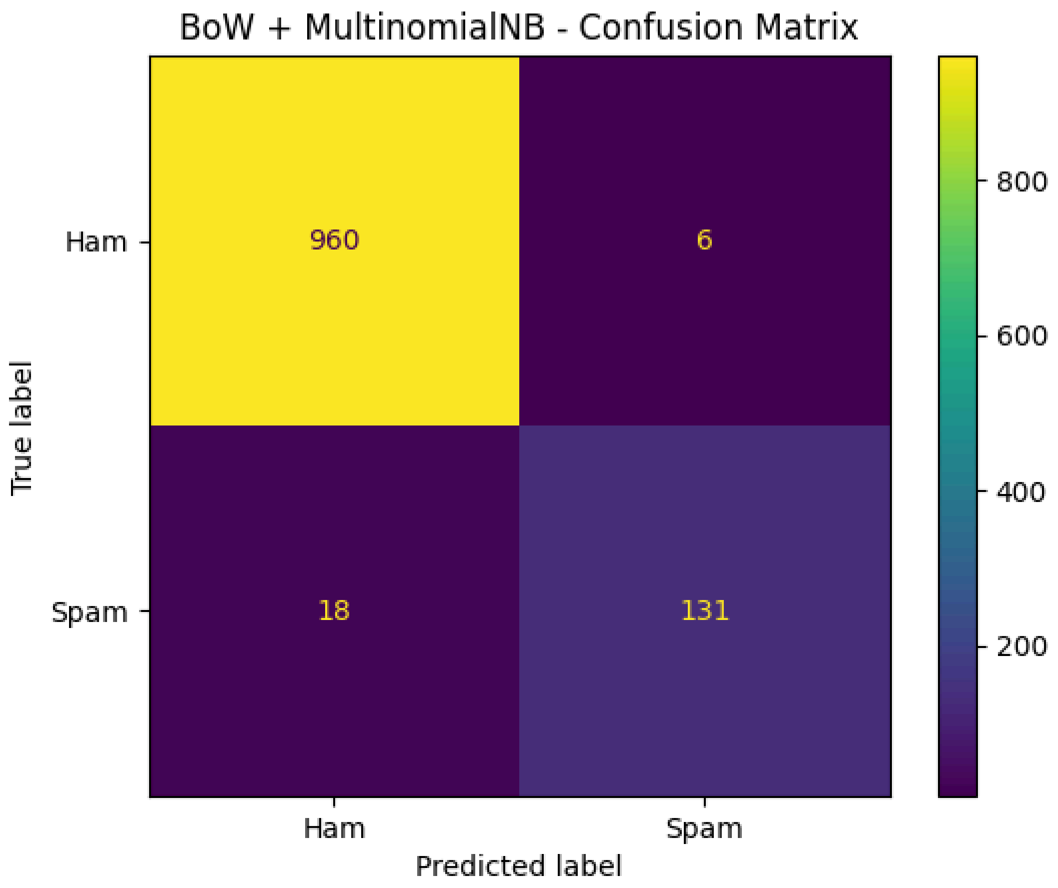
```
✅ Dataset Split Done
Train size: 4457
Test size : 1115
```

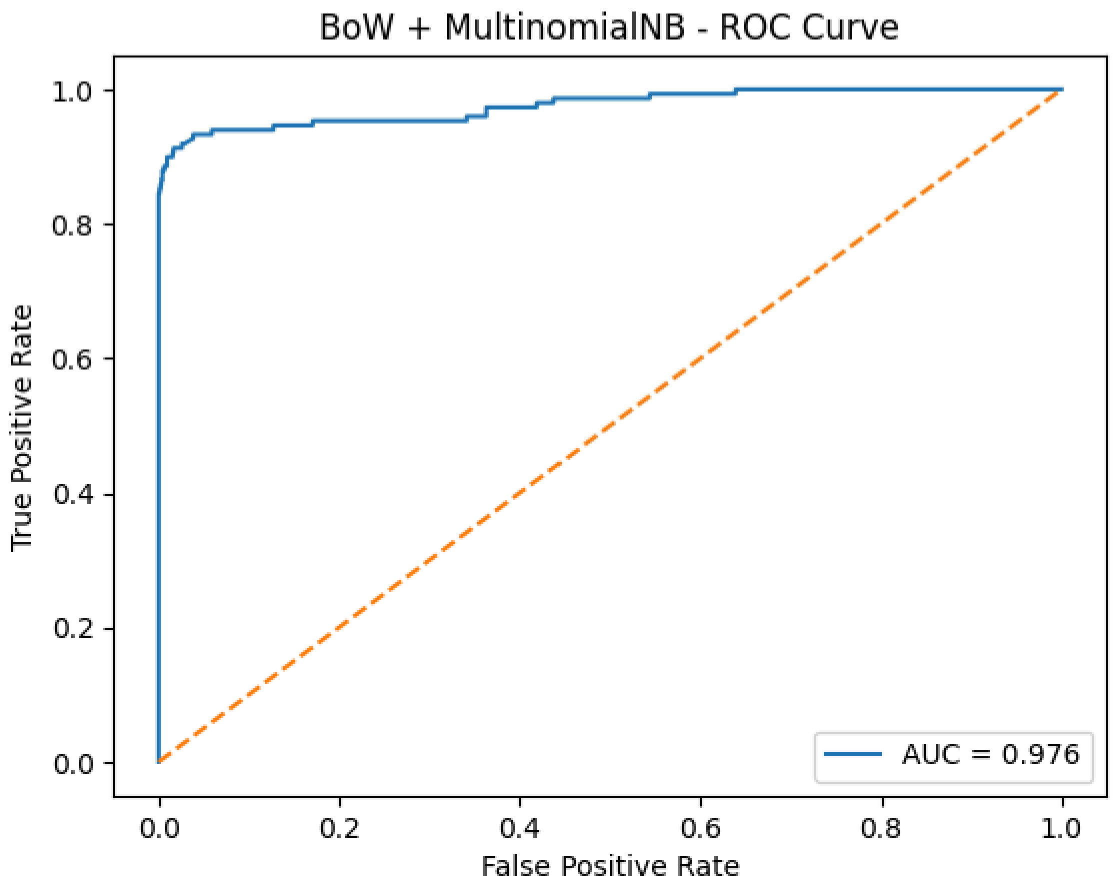
In [12]:
```
def evaluate_and_show(model, model_type, X_test_vec, y_test, title):
    """
    Prints metrics + shows Confusion Matrix and ROC immediately
    """
    if model_type == "gaussian":
        y_pred = model.predict(X_test_vec.toarray())
        y_proba = model.predict_proba(X_test_vec.toarray())[:, 1]
    else:
        y_pred = model.predict(X_test_vec)
        y_proba = model.predict_proba(X_test_vec)[:, 1]

    # metrics
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, zero_division=0)
    recall = recall_score(y_test, y_pred, zero_division=0)
    f1 = f1_score(y_test, y_pred, zero_division=0)

    print("\n========================================")
    print("MODEL:", title)
    print("========================================")
    print("Accuracy :", round(accuracy, 4))
    print("Precision:", round(precision, 4))
    print("Recall   :", round(recall, 4))
    print("F1-score :", round(f1, 4))
    print("\nClassification Report:\n", classification_report(y_test, y_pred))
    # --- confusion matrix ---
    cm = confusion_matrix(y_test, y_pred)
    disp = ConfusionMatrixDisplay(cm, display_labels=["Ham", "Spam"])
    disp.plot(values_format="d")
    plt.title(title + " - Confusion Matrix")
    plt.show()

    # --- ROC curve ---
    fpr, tpr, _ = roc_curve(y_test, y_proba)
    roc_auc = auc(fpr, tpr)

    plt.figure()
    plt.plot(fpr, tpr, label=f"AUC = {roc_auc:.3f}")
    plt.plot([0, 1], [0, 1], linestyle="--")
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title(title + " - ROC Curve")
    plt.legend(loc="lower right")
    plt.show()
```

In [13]:

```
bow = CountVectorizer(max_features=5000)

X_train_bow = bow.fit_transform(X_train)
X_test_bow = bow.transform(X_test)

mnb_bow = MultinomialNB()
mnb_bow.fit(X_train_bow, y_train)

evaluate_and_show(mnb_bow, "multinomial", X_test_bow, y_test, "BoW + MultinomialNB")
```

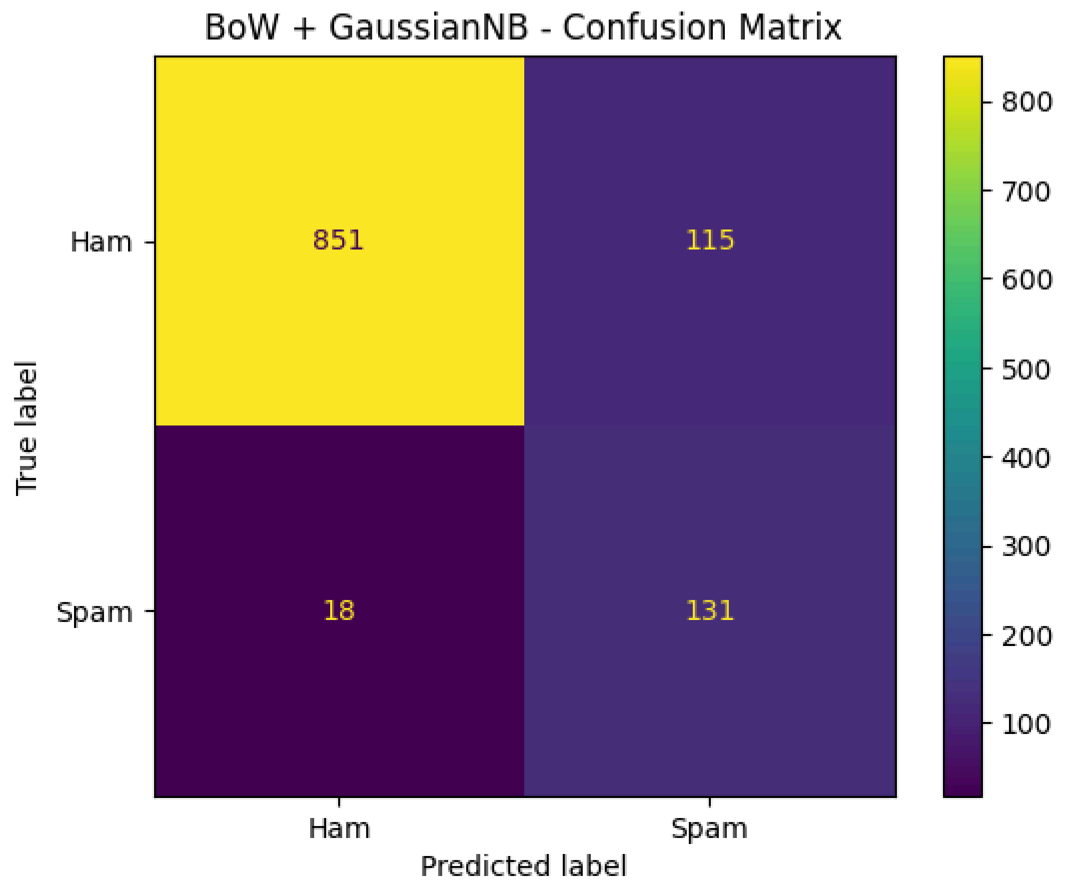
```
========================================
MODEL: BoW + MultinomialNB
========================================
Accuracy : 0.9785
Precision: 0.9562
Recall   : 0.8792
F1-score : 0.9161

Classification Report:
               precision    recall  f1-score   support

           0       0.98      0.99      0.99       966
           1       0.96      0.88      0.92       149

    accuracy                           0.98      1115
   macro avg       0.97      0.94      0.95      1115
weighted avg       0.98      0.98      0.98      1115
```

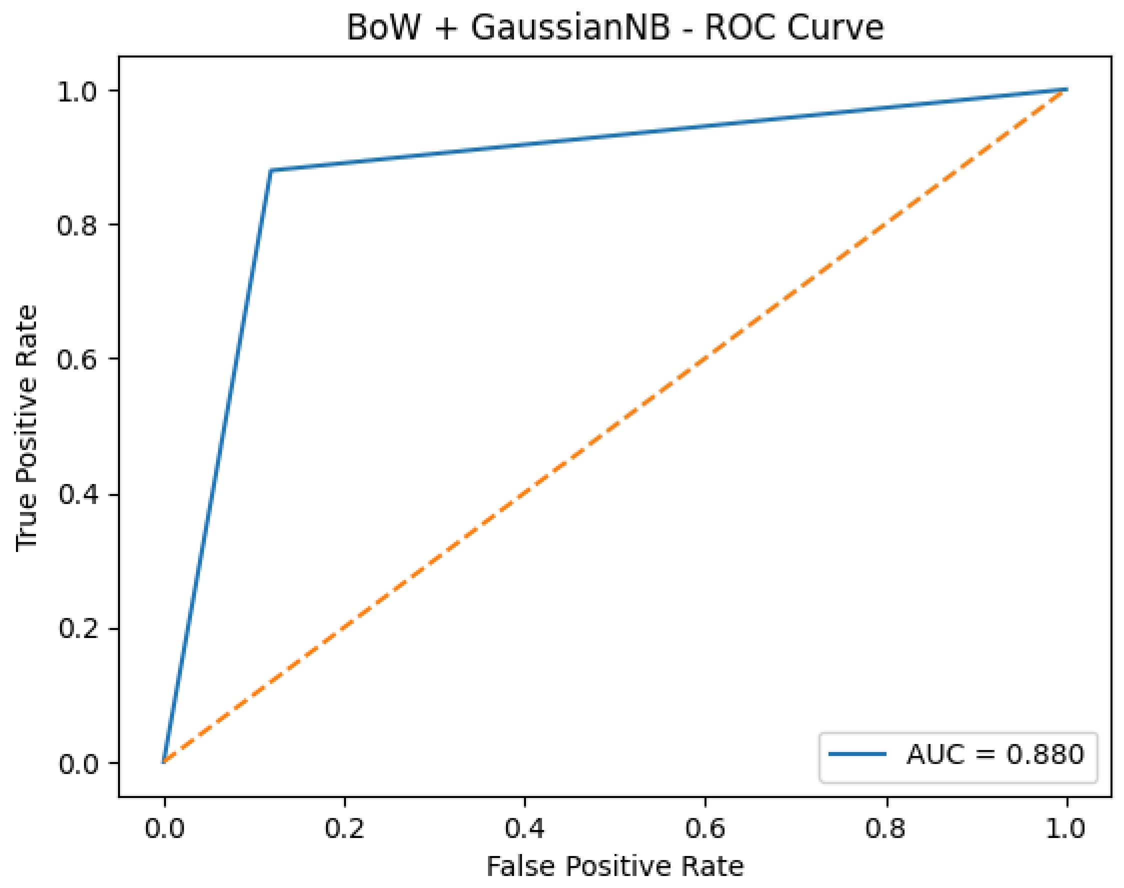

BoW + MultinomialNB - Confusion Matrix
True label
Ham
Spam
960
6
18
131
Ham
Spam
Predicted label
800
600
400
200



In [14]:
```
gnb_bow = GaussianNB()
gnb_bow.fit(X_train_bow.toarray(), y_train)

evaluate_and_show(gnb_bow, "gaussian", X_test_bow, y_test, "BoW + GaussianNB")
```

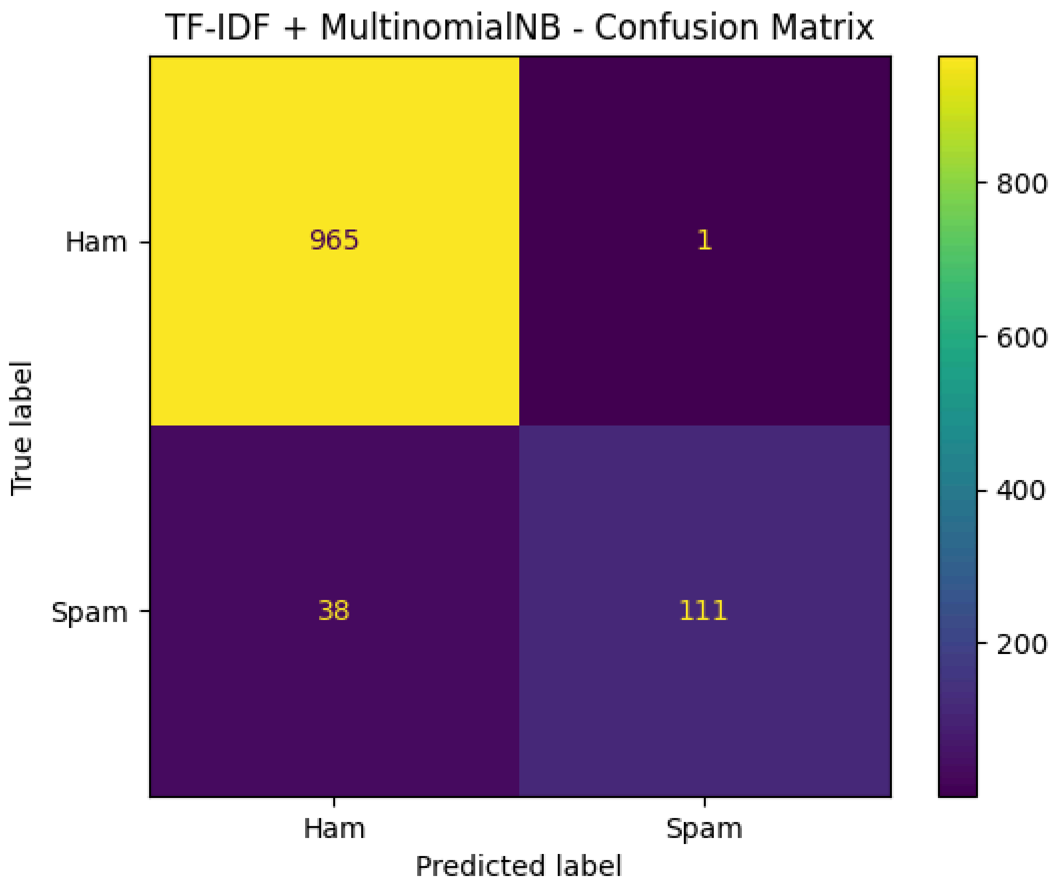
```
=======================================
MODEL: BoW + GaussianNB
=======================================
Accuracy : 0.8807
Precision: 0.5325
Recall   : 0.8792
F1-score : 0.6633

Classification Report:
               precision    recall  f1-score   support

           0       0.98      0.88      0.93       966
           1       0.53      0.88      0.66       149

    accuracy                           0.88      1115
   macro avg       0.76      0.88      0.80      1115
weighted avg       0.92      0.88      0.89      1115
```

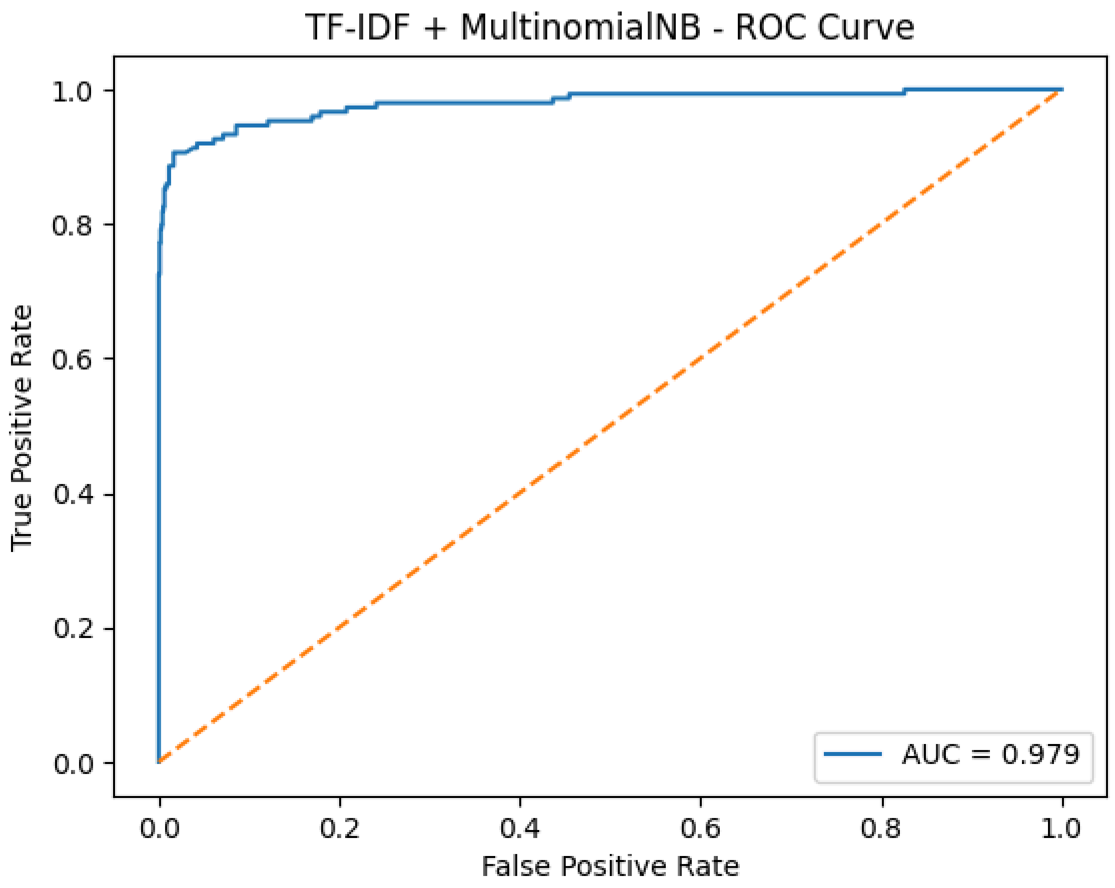

BoW + GaussianNB - Confusion Matrix
True label
Predicted label
Ham
Spam
Ham
Spam
851
115
18
131
800
700
600
500
400
300
200
100



```
In [15]: tfidf = TfidfVectorizer(max_features=5000)

         X_train_tfidf = tfidf.fit_transform(X_train)
         X_test_tfidf = tfidf.transform(X_test)

         mnb_tfidf = MultinomialNB()
         mnb_tfidf.fit(X_train_tfidf, y_train)

         evaluate_and_show(mnb_tfidf, "multinomial", X_test_tfidf, y_test, "TF-IDF + MultinomialNB")
```

```
=======================================
MODEL: TF-IDF + MultinomialNB
=======================================
Accuracy : 0.965
Precision: 0.9911
Recall   : 0.745
F1-score : 0.8506

Classification Report:
               precision    recall  f1-score   support

           0       0.96      1.00      0.98       966
           1       0.99      0.74      0.85       149

    accuracy                           0.97      1115
   macro avg       0.98      0.87      0.92      1115
weighted avg       0.97      0.97      0.96      1115
```

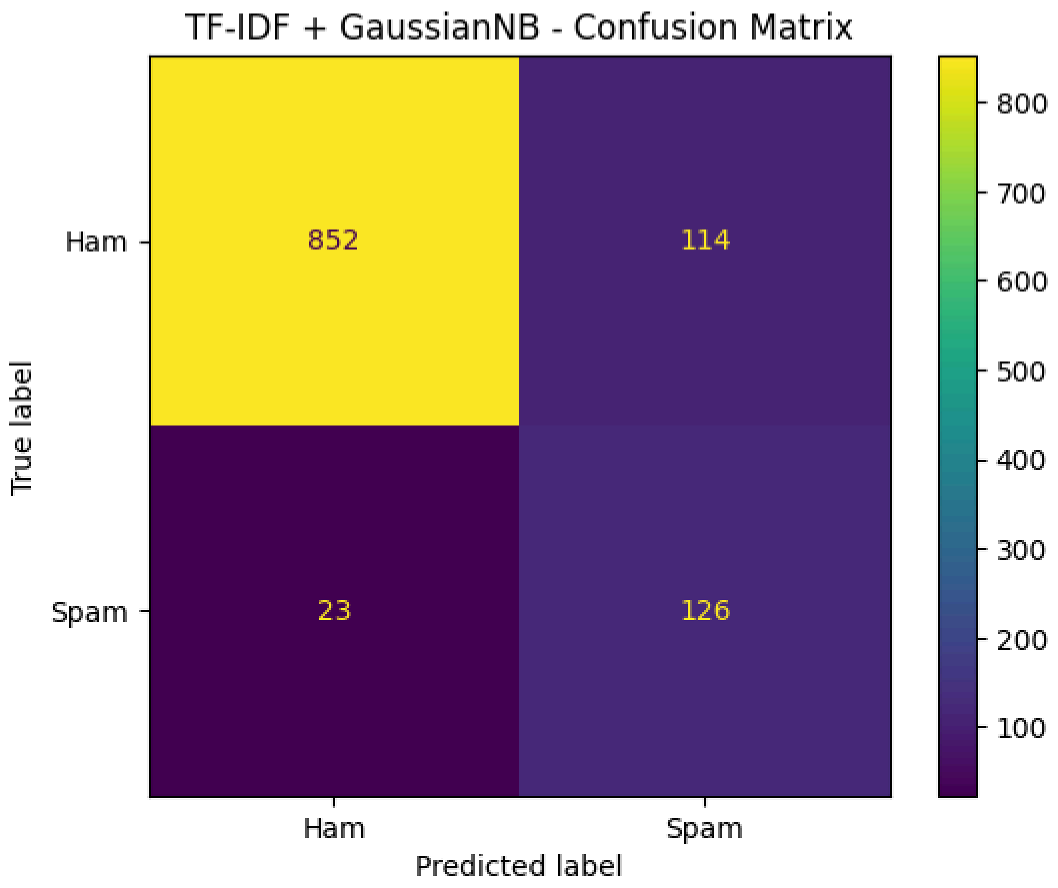

TF-IDF + MultinomialNB - Confusion Matrix
True label
Ham
Spam
965
1
38
111
Ham
Spam
Predicted label
800
600
400
200

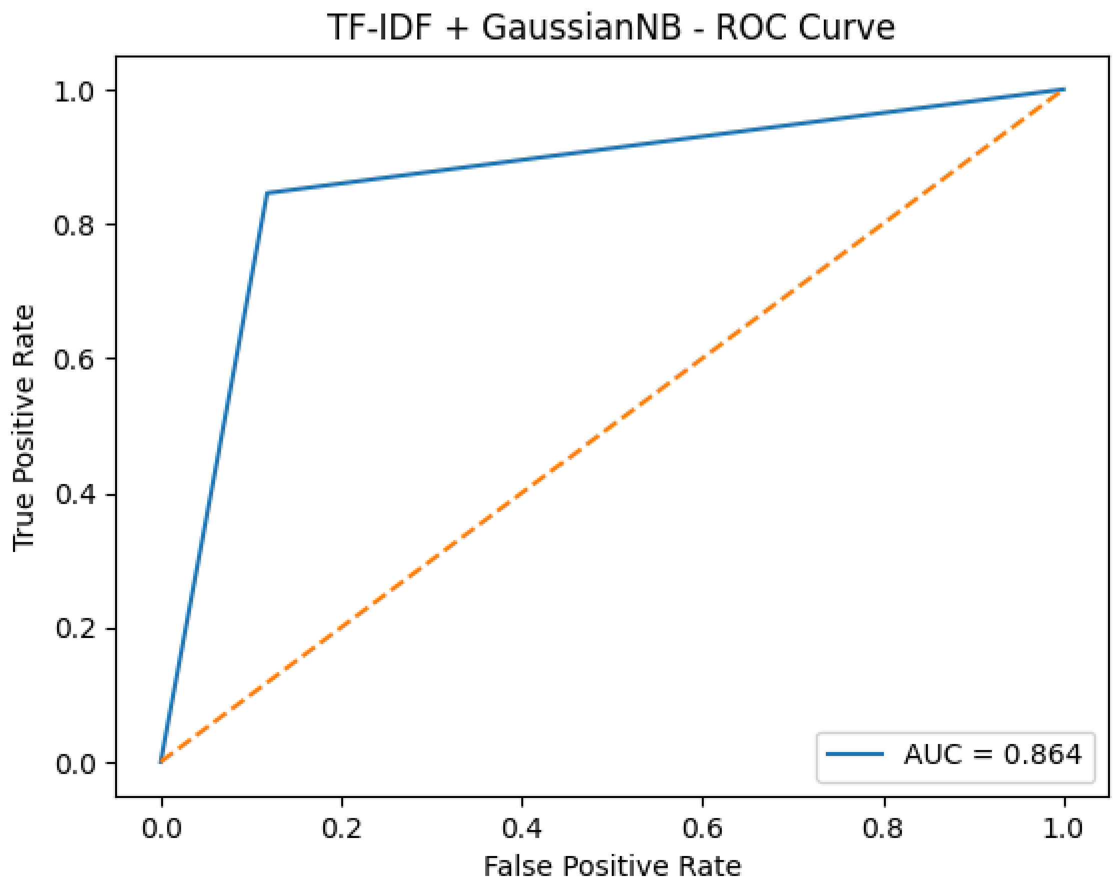


```
In [ ]:
```

```
In [16]: gnb_tfidf = GaussianNB()
         gnb_tfidf.fit(X_train_tfidf.toarray(), y_train)

         evaluate_and_show(gnb_tfidf, "gaussian", X_test_tfidf, y_test, "TF-IDF + GaussianNB")
```

```
=======================================
MODEL: TF-IDF + GaussianNB
=======================================
Accuracy : 0.8771
Precision: 0.525
Recall   : 0.8456
F1-score : 0.6478

Classification Report:
               precision    recall  f1-score   support

           0       0.97      0.88      0.93       966
           1       0.53      0.85      0.65       149

    accuracy                           0.88      1115
   macro avg       0.75      0.86      0.79      1115
weighted avg       0.91      0.88      0.89      1115
```


TF-IDF + GaussianNB - Confusion Matrix
True label
Ham
Spam
Predicted label
Ham
Spam
852
114
23
126
800
700
600
500
400
300
200
100



In [17]:
```
def predict_new_email(text):
    cleaned = clean_text(text)
    vec = tfidf.transform([cleaned])
    pred = mnb_tfidf.predict(vec)[0]
    proba = mnb_tfidf.predict_proba(vec)[0][1]
    return ("SPAM" if pred == 1 else "HAM"), proba

demo_emails = [
    "Congratulations! You won a free iPhone. Click to claim prize",
    "Hey, can we meet tomorrow at 10am for project?",
    "Urgent: Your account is blocked, verify immediately.",
    "Please submit your assignment before 5pm."
]
```

```
for mail in demo_emails:
    label, prob = predict_new_email(mail)
    print("\nEmail:", mail)
    print("Prediction:", label)
    print("Spam Probability:", round(prob, 3))
```

```
Email: Congratulations! You won a free iPhone. Click to claim prize
Prediction: SPAM
Spam Probability: 0.966

Email: Hey, can we meet tomorrow at 10am for project?
Prediction: HAM
Spam Probability: 0.003

Email: Urgent: Your account is blocked, verify immediately.
Prediction: SPAM
Spam Probability: 0.782

Email: Please submit your assignment before 5pm.
Prediction: HAM
Spam Probability: 0.084
```

```
In [ ]:
```