```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from scipy.sparse import vstack,csr_matrix

data = pd.read_csv("/content/spam.csv",encoding="latin-1")
df = data[['v1', 'v2']].rename(columns={'v1': 'label', 'v2': 'text'})
df['label'] = df['label'].map({'ham': 0, 'spam': 1})

data.head()
```


```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 5572,\n  \"fields\":
[\n    {\n      \"column\": \"v1\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"spam\",\n          \"ham\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"v2\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5169,\n
\"samples\": [\n          \"Did u download the fring app?\",\n
\"Pass dis to all ur contacts n see wat u get! Red;i'm in luv wid u.
Blue;u put a smile on my face. Purple;u r realy hot. Pink;u r so swt.
Orange;i thnk i lyk u. Green;i realy wana go out wid u. Yelow;i wnt u
bck. Black;i'm jealous of u. Brown;i miss you Nw plz giv me one
color\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Unnamed: 2\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 43,\n
\"samples\": [\n          \" GOD said\",\n          \" SHE SHUDVETOLD
U. DID URGRAN KNOW?NEWAY\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Unnamed: 3\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 10,\n
\"samples\": [\n          \" \\\\\\\"OH No! COMPETITION\\\\\\\". Who
knew\",\n          \" why to miss them\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Unnamed: 4\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n
\"GNT:-)\\\"\",\n          \" one day these two will become FREINDS
FOREVER!\\\"\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"data"}
```


```
data.tail()
```


```
{"repr_error":"0","type":"dataframe"}
```


```
def split_into_two(text):
    words = str(text).split()
    middle = len(words) // 2
```

```
    return " ".join(words[:middle]), " ".join(words[middle:])

df[['view1', 'view2']] = df['text'].apply(lambda x:
pd.Series(split_into_two(x)))

df['is_labeled'] = False
df.loc[:int(len(df) * 0.2), 'is_labeled'] = True

vectorizer = CountVectorizer()
vectorizer.fit(pd.concat([df['view1'], df['view2']]))

X1_labeled = vectorizer.transform(df[df.is_labeled]['view1'])
X2_labeled = vectorizer.transform(df[df.is_labeled]['view2'])
y_labeled = df[df.is_labeled]['label'].values

X1_unlabeled = vectorizer.transform(df[~df.is_labeled]['view1'])
X2_unlabeled = vectorizer.transform(df[~df.is_labeled]['view2'])

model1 = MultinomialNB()
model2 = MultinomialNB()

for round_num in range(3):
print(f"\n Round {round_num + 1}")
model1.fit(X1_labeled, y_labeled)
model2.fit(X2_labeled, y_labeled)

probs1 = model1.predict_proba(X1_unlabeled)
probs2 = model2.predict_proba(X2_unlabeled)

confident_indexes = []
for i, (p1, p2) in enumerate(zip(probs1, probs2)):
if max(p1) > 0.9 and max(p2) > 0.9 and np.argmax(p1) == np.argmax(p2):

confident_indexes.append(i)
if not confident_indexes:
print(" No confident samples this round.")
break

  File "/tmp/ipython-input-16-1067755844.py", line 2
    print(f"\n Round {round_num + 1}")
    ^
IndentationError: expected an indented block after 'for' statement on
line 1


for round_num in range(3):
    print(f"\n Round {round_num + 1}")
    model1.fit(X1_labeled, y_labeled)
    model2.fit(X2_labeled, y_labeled)

    probs1 = model1.predict_proba(X1_unlabeled)
```

```
    probs2 = model2.predict_proba(X2_unlabeled)

    confident_indexes = []
    for i, (p1, p2) in enumerate(zip(probs1, probs2)):
        if max(p1) > 0.9 and max(p2) > 0.9 and np.argmax(p1) ==
np.argmax(p2):
            confident_indexes.append(i)

    if not confident_indexes:
        print(" No confident samples this round.")
        break

    # Add confident samples to labeled set
    X1_labeled = vstack([X1_labeled, X1_unlabeled[confident_indexes]])
    X2_labeled = vstack([X2_labeled, X2_unlabeled[confident_indexes]])
    y_labeled = np.concatenate([y_labeled,
np.argmax(probs1[confident_indexes], axis=1)])

    # Remove confident samples from unlabeled set
    all_unlabeled_indexes = np.arange(X1_unlabeled.shape[0])
    unlabeled_to_keep = np.setdiff1d(all_unlabeled_indexes,
confident_indexes)
    X1_unlabeled = X1_unlabeled[unlabeled_to_keep]
    X2_unlabeled = X2_unlabeled[unlabeled_to_keep]


 Round 1

 Round 2

 Round 3


new_headline = "Technology drives future"
v1, v2 =split_into_two(new_headline)
X_new1 = vectorizer.transform([v1])
X_new2 = vectorizer.transform([v2])
p1 = model1.predict(X_new1)[0]
p2 = model2.predict(X_new2)[0]
print("\n Final Prediction:")
print("Model 1 says:", p1)
print("Model 2 says:",p2)


 Final Prediction:
Model 1 says: 0
Model 2 says: 0
```