```
import nltk
import math
nltk.download('stopwords')
import random
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, classification_report, log_loss
import pandas as pd
import numpy as np

df = pd.read_csv('../content/best_selling_switch_games.csv', header=0, usecols=[0,3], encoding='latin-1')
print('rows and columns:', df.shape)

X = df.title
y = df.developer

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8, random_state=1234)
X_train.shape

stopwords = set(stopwords.words('english'))
vectorizer = TfidfVectorizer(stop_words=list(stopwords))
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)

naive_bayes = MultinomialNB()
naive_bayes.fit(X_train, y_train)

prior_p = sum(y_train == 1)/len(y_train)

naive_bayes.class_log_prior_[1]
naive_bayes.feature_log_prob_

pred = naive_bayes.predict(X_test)
print(confusion_matrix(y_test, pred))

print('accuracy score: ', accuracy_score(y_test, pred))

print('\nprecision score (not dev): ', precision_score(y_test, pred, pos_label=0, average='micro'))
print('precision score (dev): ', precision_score(y_test, pred, average='micro'))

print('\nrecall score: (not dev)', recall_score(y_test, pred, pos_label=0, average='micro'))
print('recall score: (dev)', recall_score(y_test, pred, average='micro'))

print('\nf1 score: ', f1_score(y_test, pred, average='micro'))

print(classification_report(y_test, pred))

classifier = LogisticRegression(solver='lbfgs', class_weight='balanced')
classifier.fit(X_train, y_train)

pred = classifier.predict(X_test)
print('accuracy score: ', accuracy_score(y_test, pred))
print('precision score: ', precision_score(y_test, pred, average='micro'))
print('recall score: ', recall_score(y_test, pred, average='micro'))
print('f1 score: ', f1_score(y_test, pred, average='micro'))
probs = classifier.predict_proba(X_test)

classifier = MLPClassifier(solver='adam', alpha=1e-5,
                   hidden_layer_sizes=(15, 2), random_state=1)
classifier.fit(X_train, y_train)

print('accuracy score: ', accuracy_score(y_test, pred))
print('precision score: ', precision_score(y_test, pred, average='micro'))
print('recall score: ', recall_score(y_test, pred, average='micro'))
print('f1 score: ', f1_score(y_test, pred, average='micro'))
```