```
X_train = ["This was really awesome an awesome movie",
        "Great movie! Ilikes it a lot",
        "Happy Ending! Awesome Acting by hero",
        "loved it!",
        "Bad not upto the mark",
        "Could have been better",
        "really Dissapointed by the movie"]
# X_test = "it was really awesome and really dissptd"

y_train = ["positive","positive","positive","positive","negative","negative","negative"] # 1- Positive class, 0- negative class




X_train # Reviews


# Cleaning of the data


# Tokenize

# "I am a python dev" -> ["I", "am", "a", "python", "dev"]


from nltk.tokenize import RegexpTokenizer

# NLTK -> Tokenize -> RegexpTokenizer


# Stemming

# "Playing" -> "Play"

# "Working" -> "Work"


from nltk.stem.porter import PorterStemmer

# NLTK -> Stem -> Porter -> PorterStemmer
```

```
from nltk.corpus import stopwords
# NLTK -> Corpus -> stopwords

# Downloading the stopwords
import nltk
nltk.download('stopwords')

tokenizer = RegexpTokenizer(r"\w+")
en_stopwords = set(stopwords.words('english'))
ps = PorterStemmer()

def getCleanedText(text):
  text = text.lower()

  # tokenizing
  tokens = tokenizer.tokenize(text)
  new_tokens = [token for token in tokens if token not in en_stopwords]
  stemmed_tokens = [ps.stem(tokens) for tokens in new_tokens]
  clean_text = " ".join(stemmed_tokens)
  return clean_text

# Input from the user

X_test = ["it was bad"]

X_clean = [getCleanedText(i) for i in X_train]
xt_clean = [getCleanedText(i) for i in X_test]

X_clean

# Data before cleaning
```

```
'''
X_train = ["This was awesome an awesome movie",
        "Great movie! Ilikes it a lot",
        "Happy Ending! Awesome Acting by hero",
        "loved it!",
        "Bad not upto the mark",
        "Could have been better",
        "Dissapointed by the movie"]
'''

# Vectorize

from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(ngram_range = (1,2))
# "I am PyDev" -> "i am", "am Pydev"

X_vec = cv.fit_transform(X_clean).toarray()

X_vec

print(cv.get_feature_names())

Xt_vect = cv.transform(xt_clean).toarray()

Xt_vect

# Multinomial Naive Bayes

from sklearn.naive_bayes import MultinomialNB
```

```
mn = MultinomialNB()


mn.fit(X_vec, y_train)


y_pred = mn.predict(Xt_vect)


y_pred
```