```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(sublinear_tf=True,
                        min_df=5, norm='l2',
                        ngram_range=(1, 2), stop_words='english')
features = tfidf.fit_transform(concated[:40000].name).toarray()
df1 = pd.DataFrame(features, columns=tfidf.get_feature_names())


#Dimenionality reduction. Only using the 100 best features per category
from sklearn.decomposition import PCA
pca = PCA(n_components= 100,random_state=3)
X = pca.fit_transform(X)


X_test = tfidf.fit_transform(names['Product Name']).toarray()
X_test = pca.fit_transform(X_test)
y_pred = model.predict(X_test)
```

multiple class

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n_estimators=100, max_features=3, oob_score=True)
model.fit(X_train, y_train)

labels = model.classes_

from sklearn.metrics import confusion_matrix
import numpy as np
pred_train = np.argmax(model.oob_decision_function_,axis=1)
pred_label = [labels[pred_train[i]] for i in range(len(pred_train))]

print(
"OOB accuracy is",
model.oob_score_, "\n",
"OOB Confusion Matrix", "\n",
pd.DataFrame(confusion_matrix(y_train, pred_label))
)

from sklearn.metrics import auc, roc_curve, classification_report, roc_auc_score
roc_auc_score(y_train, model.oob_decision_function_, multi_class='ovr')
```