3107 – JAWAHAR ENGINEERING COLLEGE

Subject Title - AI 101- Artificial Intelligence: Phase-4
Project Title – Building a Smarter AI Powered Classifier

Team Member – 310721205002: S. Muthunivas Pandi

S.No.	NaanMudhalvan ID	Position	Name	Department
		Faculty Mentor	V. Nivaskumar	
1.	au310721106001	Team Head	Junia Susheela Shalom	ECE
2.	au310721205001	Team member	M. Hariragavan	IT
3.	au310721205002	Team Member	S. Muthunivas Pandi	IT

1. Model Building

In [59]:

```
X = tfidf.fit transform(dataset['transformed text']).toarray()
```

In [60]:
```
# appending the num_character column to X
X = np.hstack((X,dataset['num_characters'].values.reshape(-1, 1)))
```

In [61]:
```
X.shape
```

Out[61]:
```
(5169, 3001)
```

In [62]:
```
y = dataset['type'].values
y
```

Out[62]:
```
array([0, 0, 1, ..., 0, 0, 0])
```

In [63]:
```
from sklearn.model_selection import train_test_split
```

In [64]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)
```

In [65]:
```
from sklearn.naive_bayes import GaussianNB, MultinomialNB, BernoulliNB
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score
```

In [66]:
```
gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()
```

In [67]:
```
gnb.fit(X_train, y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test, y_pred1))
print(confusion_matrix(y_test, y_pred1))
print(precision_score(y_test, y_pred1))
```

```
0.8907156673114119
[[807  89]
 [ 24 114]]
0.5615763546798029
```

In [68]:
```
mnb.fit(X_train, y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test, y_pred2))
print(confusion_matrix(y_test, y_pred2))
print(precision_score(y_test, y_pred2))
```

```
0.9410058027079303
[[896   0]
 [ 61  77]]
1.0
```

In [69]:
```
bnb.fit(X_train, y_train)
y_pred3 = bnb.predict(X_test)
print(accuracy_score(y_test, y_pred3))
print(confusion_matrix(y_test, y_pred3))
print(precision_score(y_test, y_pred3))
```

```
0.9835589941972921
[[895   1]
 [ 16 122]]
0.991869918699187
```

In [70]:
```
## tfidf --> MNB
```

In [71]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
```

In [72]:
```
svc = SVC(kernel = 'sigmoid', gamma = 1.0)
knc = KNeighborsClassifier()
mnb = MultinomialNB()
dtc = DecisionTreeClassifier(max_depth=5)
lrc = LogisticRegression(solver = 'liblinear', penalty = 'l1')
rfc = RandomForestClassifier(n_estimators=50, random_state=2)
abc = AdaBoostClassifier(n_estimators=50, random_state=2)
bc = BaggingClassifier(n_estimators=50, random_state=2)
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
gbdt = GradientBoostingClassifier(n_estimators=50, random_state=2)
xgb = XGBClassifier(n_estimators=50, random_state=2)
```

In [73]:
```
clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB' : mnb,
    'DT' : dtc,
    'LR' : lrc,
    'RF' : rfc,
    'Adaboost' : abc,
    'BgC' : bc,
    'ETC' : etc,
    'GBDT' : gbdt,
    'xgb' : xgb
}
```

In [74]:
```
def train_classifier(clf, X_train, y_train, X_test, y_test):
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    return accuracy, precision
```

In [75]:
```
train_classifier(svc, X_train, y_train, X_test, y_test)
```

```
C:\Users\shalo\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\metrics\_classi
fication.py:1469: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 due to n
o predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Out[75]:
```
(0.8665377176015474, 0.0)
```

In [76]:
```
accuracy_scores = []
precision_scores = []
```

```
for name, clf in clfs.items():
    current_accuracy, current_precision = train_classifier(clf, X_train, y_train, X_test, y_te
    print("For ", name)
    print("Accuracy - ", current_accuracy)
    print("Precision - ", current_precision)

    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
C:\Users\shalo\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\metrics\_classi
fication.py:1469: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 due to n
o predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
For  SVC
Accuracy -  0.8665377176015474
Precision -  0.0
For  KN
Accuracy -  0.9332688588007737
Precision -  0.822429906542056
For  NB
Accuracy -  0.9410058027079303
Precision -  1.0
For  DT
Accuracy -  0.9439071566731141
Precision -  0.8773584905660378
For  LR
Accuracy -  0.9613152804642167
Precision -  0.9622641509433962
For  RF
Accuracy -  0.9690522243713733
Precision -  0.9818181818181818
For  Adaboost
Accuracy -  0.9642166344294004
Precision -  0.9316239316239316
For  BgC
Accuracy -  0.9661508704061895
Precision -  0.8992248062015504
For  ETC
Accuracy -  0.9787234042553191
Precision -  0.9754098360655737
For  GBDT
Accuracy -  0.9506769825918762
Precision -  0.9306930693069307
For  xgb
Accuracy -  0.9690522243713733
Precision -  0.9416666666666667
```

In [77]:
```
performance_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy':accuracy_scores, 'Precision'
```

In [78]:
```
performance_df
```

Out[78]:

	Algorithm	Accuracy	Precision
2	NB	0.941006	1.000000
5	RF	0.969052	0.981818
8	ETC	0.978723	0.975410
4	LR	0.961315	0.962264
10	xgb	0.969052	0.941667
6	Adaboost	0.964217	0.931624
9	GBDT	0.950677	0.930693

7	BgC	0.966151	0.899225
3	DT	0.943907	0.877358
1	KN	0.933269	0.822430
0	SVC	0.866538	0.000000

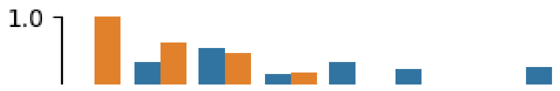
In [79]:
```
performance_df1 = pd.melt(performance_df, id_vars = "Algorithm")
```

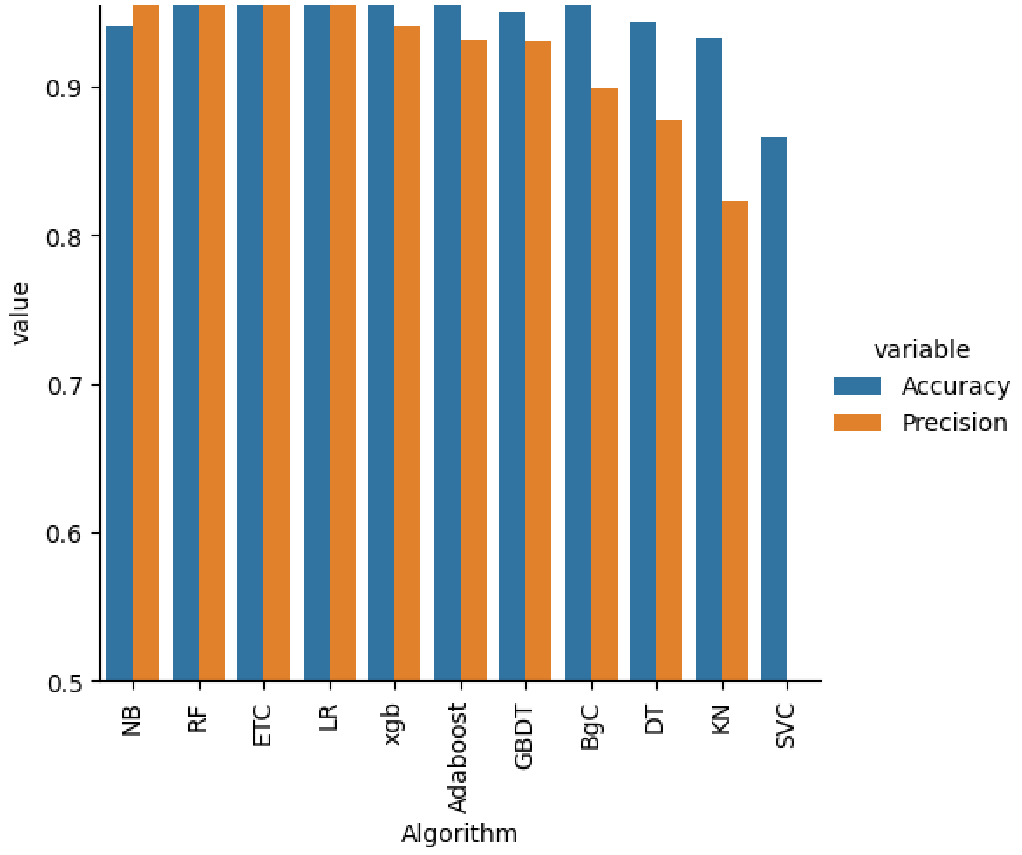
In [80]:
```
performance_df1
```

Out[80]:

	Algorithm	variable	value
0	NB	Accuracy	0.941006
1	RF	Accuracy	0.969052
2	ETC	Accuracy	0.978723
3	LR	Accuracy	0.961315
4	xgb	Accuracy	0.969052
5	Adaboost	Accuracy	0.964217
6	GBDT	Accuracy	0.950677
7	BgC	Accuracy	0.966151
8	DT	Accuracy	0.943907
9	KN	Accuracy	0.933269
10	SVC	Accuracy	0.866538
11	NB	Precision	1.000000
12	RF	Precision	0.981818
13	ETC	Precision	0.975410
14	LR	Precision	0.962264
15	xgb	Precision	0.941667
16	Adaboost	Precision	0.931624
17	GBDT	Precision	0.930693
18	BgC	Precision	0.899225
19	DT	Precision	0.877358
20	KN	Precision	0.822430
21	SVC	Precision	0.000000

In [81]:
```
sns.catplot(x = 'Algorithm', y = 'value',
               hue = 'variable', data = performance_df1, kind = 'bar', height = 5)
plt.ylim(0.5, 1.0)
plt.xticks(rotation = 'vertical')
plt.show()
```





In [82]:
```
#model improve
#1. Change the max features parameter of TfIdf
```

In [83]:
```
temp_df = pd.DataFrame({'Algorithm':clfs.keys(), 'Accuracy_max_ft_3000': accuracy_scores, 'Pre
```

In [84]:
```
performance_df.merge(temp_df, on='Algorithm')
```

Out[84]:

	Algorithm	Accuracy	Precision	Accuracy_max_ft_3000	Precision_max_ft_3000
0	NB	0.941006	1.000000	0.941006	1.000000
1	RF	0.969052	0.981818	0.969052	0.981818
2	ETC	0.978723	0.975410	0.978723	0.975410
3	LR	0.961315	0.962264	0.961315	0.962264
4	xgb	0.969052	0.941667	0.969052	0.941667
5	Adaboost	0.964217	0.931624	0.964217	0.931624
6	GBDT	0.950677	0.930693	0.950677	0.930693
7	BgC	0.966151	0.899225	0.966151	0.899225
8	DT	0.943907	0.877358	0.943907	0.877358
9	KN	0.933269	0.822430	0.933269	0.822430
10	SVC	0.866538	0.000000	0.866538	0.000000

In [88]:
```
# Voting Classifier
svc = SVC(kernel='sigmoid',gamma = 1.0, probability=True)
mnb = MultinomialNB()
```

```
etc = ExtraTreesClassifier(n_estimators = 50, random_state=2)

from sklearn.ensemble import VotingClassifier
```

In [90]:
```
voting = VotingClassifier(estimators=[('svm', svc), ('nb', mnb), ('et', etc)],voting = 'soft')
```

In [91]:
```
voting.fit(X_train, y_train)
```

Out[91]:
```
VotingClassifier(estimators=[('svm',
                              SVC(gamma=1.0, kernel='sigmoid',
                                  probability=True)),
                             ('nb', MultinomialNB()),
                             ('et',
                              ExtraTreesClassifier(n_estimators=50,
                                                   random_state=2))],
                 voting='soft')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [92]:
```
y_pred = voting.predict(X_test)
print("Accuracy", accuracy_score(y_test, y_pred))
print("Precision", precision_score(y_test, y_pred))
```

```
Accuracy 0.941972920696325
Precision 1.0
```

In [97]:
```
#Applying stacking
estimators=[('svm', svc), ('nb', mnb), ('et', etc)]
final_estimator = RandomForestClassifier()
```

In [98]:
```
from sklearn.ensemble import StackingClassifier
```

In [99]:
```
clf = StackingClassifier(estimators=estimators, final_estimator=final_estimator)
```

In [100…
```
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
print("Accuracy", accuracy_score(y_test, y_pred))
print("Precision", precision_score(y_test, y_pred))
```

```
Accuracy 0.9748549323017408
Precision 0.9307692307692308
```