import pandas as pd  
import numpy as np  
from matplotlib import pyplot as plt  
import seaborn as sns  
  
# Scikit-learn  
from sklearn.model\_selection import train\_test\_split  
from sklearn.preprocessing import LabelEncoder  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.svm import SVC  
from sklearn.naive\_bayes import BernoulliNB  
from sklearn.neural\_network import MLPClassifier  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.metrics import confusion\_matrix, accuracy\_score, f1\_score  
  
plt.rcParams["figure.figsize"] = [16, 10]

# Loading Data

fake = pd.read\_csv('DATASET/fusers.csv')  
legit = pd.read\_csv('DATASET/users.csv')

fake.head()

id

name

screen\_name

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

created\_at

url

...

profile\_background\_image\_url

profile\_background\_color

profile\_link\_color

utc\_offset

protected

verified

description

updated

dataset

label

0

370098498

pirfectmoses

pirfectmoses

24

4

588

16

0

Thu Sep 08 13:20:35 +0000 2011

NaN

...

http://a0.twimg.com/profile\_background\_images/...

C6E2EE

1F98C7

NaN

NaN

NaN

NaN

2/14/2015 10:40

INT

fake

1

37384589

SAK Nair

bsknair1967

656

57

693

597

0

Sun May 03 07:35:13 +0000 2009

NaN

...

http://a0.twimg.com/images/themes/theme1/bg.png

C0DEED

0084B4

NaN

NaN

NaN

in Kuwait with my buetiful family.

2/14/2015 10:40

INT

fake

2

72110028

Deepak

dedjven

1234

15

104

1150

0

Sun Sep 06 19:50:08 +0000 2009

NaN

...

http://a0.twimg.com/images/themes/theme14/bg.gif

131516

9999

-39600.0

NaN

NaN

NaN

2/14/2015 10:40

INT

fake

3

82885728

Marcos Vinicius

BrowAlves

573

14

227

530

0

Fri Oct 16 14:02:48 +0000 2009

NaN

...

http://a0.twimg.com/profile\_background\_images/...

C0DEED

0084B4

NaN

NaN

NaN

NaN

2/14/2015 10:40

INT

fake

4

110120789

Shri Kant Kanaujia

kanaujiask

675

18

519

653

0

Sun Jan 31 12:08:41 +0000 2010

NaN

...

http://a0.twimg.com/images/themes/theme1/bg.png

C0DEED

0084B4

19800.0

NaN

NaN

NaN

2/14/2015 10:40

INT

fake

5 rows × 35 columns

legit.head()

id

name

screen\_name

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

created\_at

url

...

profile\_background\_image\_url

profile\_background\_color

profile\_link\_color

utc\_offset

protected

verified

description

updated

dataset

label

0

3610511

Davide Dellacasa

braddd

20370

5470

2385

145

52

Fri Apr 06 10:58:22 +0000 2007

http://braddd.tumblr.com

...

http://a0.twimg.com/profile\_background\_images/...

BADFCD

FF0000

3600.0

NaN

NaN

Founder of http://www.screenweek.it & http://w...

2/14/2015 10:54

E13

legit

1

5656162

Simone Economo

eKoeS

3131

506

381

9

40

Mon Apr 30 15:08:42 +0000 2007

http://www.lineheight.net/

...

http://a0.twimg.com/images/themes/theme1/bg.png

C0DEED

0084B4

3600.0

NaN

NaN

BSc degree (cum laude) in Computer Engineering...

2/14/2015 10:54

E13

legit

2

5682702

tacone

tacone\_

4024

264

87

323

16

Tue May 01 11:53:40 +0000 2007

http://t.co/LKrl1dZE

...

http://a0.twimg.com/profile\_background\_images/...

1A1B1F

2FC2EF

3600.0

NaN

NaN

Cogito ergo bestemmio.

2/14/2015 10:54

E13

legit

3

6067292

alesaura

alesstar

40586

640

622

1118

32

Tue May 15 16:55:16 +0000 2007

http://alesstar.wordpress.com/

...

http://a0.twimg.com/images/themes/theme4/bg.gif

0099B9

0099B9

3600.0

NaN

NaN

Se la vita ti dà sarde, scapocciale!

2/14/2015 10:54

E13

legit

4

6015122

Angelo

PerDiletto

2016

62

64

13

0

Sun May 13 19:52:00 +0000 2007

http://www.flickr.com/per\_diletto

...

http://a0.twimg.com/images/themes/theme18/bg.gif

ACDED6

38543

3600.0

NaN

NaN

Je me souviens

2/14/2015 10:54

E13

legit

5 rows × 35 columns

# Removing unnecessary columns

fake = fake.drop(columns=["id", "name", "screen\_name", "created\_at", "lang", "location", "default\_profile", "default\_profile\_image", "geo\_enabled", "profile\_image\_url", "profile\_banner\_url", "profile\_use\_background\_image", "profile\_background\_image\_url\_https", "profile\_text\_color", "profile\_image\_url\_https", "profile\_sidebar\_border\_color", "profile\_background\_tile", "profile\_sidebar\_fill\_color", "profile\_background\_image\_url", "profile\_background\_color", "profile\_link\_color", "utc\_offset", "protected", "verified", "dataset", "updated", "description", "url"], axis=1)  
legit = legit.drop(columns=["id", "name", "screen\_name", "created\_at", "lang", "location", "default\_profile", "default\_profile\_image", "geo\_enabled", "profile\_image\_url", "profile\_banner\_url", "profile\_use\_background\_image", "profile\_background\_image\_url\_https", "profile\_text\_color", "profile\_image\_url\_https", "profile\_sidebar\_border\_color", "profile\_background\_tile", "profile\_sidebar\_fill\_color", "profile\_background\_image\_url", "profile\_background\_color", "profile\_link\_color", "utc\_offset", "protected", "verified", "dataset", "updated", "description", "url"], axis=1)

fake.head()

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

time\_zone

label

0

24

4

588

16

0

NaN

fake

1

656

57

693

597

0

NaN

fake

2

1234

15

104

1150

0

International Date Line West

fake

3

573

14

227

530

0

NaN

fake

4

675

18

519

653

0

New Delhi

fake

legit.head()

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

time\_zone

label

0

20370

5470

2385

145

52

Rome

legit

1

3131

506

381

9

40

Rome

legit

2

4024

264

87

323

16

Rome

legit

3

40586

640

622

1118

32

Rome

legit

4

2016

62

64

13

0

Rome

legit

# Changing all NaN to 0

fake = fake.fillna(0)  
legit = legit.fillna(0)

fake.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1337 entries, 0 to 1336  
Data columns (total 7 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 statuses\_count 1337 non-null int64   
 1 followers\_count 1337 non-null int64   
 2 friends\_count 1337 non-null int64   
 3 favourites\_count 1337 non-null int64   
 4 listed\_count 1337 non-null int64   
 5 time\_zone 1337 non-null object  
 6 label 1337 non-null object  
dtypes: int64(5), object(2)  
memory usage: 73.2+ KB

legit.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1481 entries, 0 to 1480  
Data columns (total 7 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 statuses\_count 1481 non-null int64   
 1 followers\_count 1481 non-null int64   
 2 friends\_count 1481 non-null int64   
 3 favourites\_count 1481 non-null int64   
 4 listed\_count 1481 non-null int64   
 5 time\_zone 1481 non-null object  
 6 label 1481 non-null object  
dtypes: int64(5), object(2)  
memory usage: 81.1+ KB

# Creating merged dataset from legit and fake profile dataset.

data = [fake, legit]  
dataset = pd.concat(data)  
dataset.head()

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

time\_zone

label

0

24

4

588

16

0

0

fake

1

656

57

693

597

0

0

fake

2

1234

15

104

1150

0

International Date Line West

fake

3

573

14

227

530

0

0

fake

4

675

18

519

653

0

New Delhi

fake

# Encoding Features with "Object" Datatype

le = LabelEncoder()

dataset['time\_zone'] = le.fit\_transform(dataset['time\_zone'].astype(str))  
dataset['label'] = le.fit\_transform(dataset['label'].astype(str))  
dataset.head()

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

time\_zone

label

0

24

4

588

16

0

0

0

1

656

57

693

597

0

0

0

2

1234

15

104

1150

0

17

0

3

573

14

227

530

0

0

0

4

675

18

519

653

0

24

0

# Shuffling Data Instances to Mix Up the Fake and Legit Instances

dataset = dataset.sample(frac=1)  
dataset.head()

statuses\_count

followers\_count

friends\_count

favourites\_count

listed\_count

time\_zone

label

770

38

17

288

2

0

0

0

35

59

24

619

0

0

0

0

556

48

23

389

0

0

0

0

197

34098

13890

12773

85

486

4

1

343

770

42

149

28

0

2

1

# Splitting Label and Features

y = dataset['label']  
X = dataset.drop(columns='label')

# Splitting Dataset into Train and Test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Initializing Models

rfc = RandomForestClassifier()  
svm = SVC()  
bnb = BernoulliNB()  
mlp = MLPClassifier(max\_iter=250)  
dt = DecisionTreeClassifier()

# Training the Models

rfc.fit(X\_train, y\_train)  
svm.fit(X\_train, y\_train)  
bnb.fit(X\_train, y\_train)  
mlp.fit(X\_train, y\_train)  
dt.fit(X\_train, y\_train)

DecisionTreeClassifier()

# Predicting with the Models

rfc\_pred = rfc.predict(X\_test)  
svm\_pred = svm.predict(X\_test)  
bnb\_pred = bnb.predict(X\_test)  
mlp\_pred = mlp.predict(X\_test)  
dt\_pred = dt.predict(X\_test)

# Performance Metrics

fgraph = pd.DataFrame([[0, len(y\_test == 0), len(rfc\_pred == 0), len(svm\_pred == 0), len(bnb\_pred == 0), len(mlp\_pred == 0), len(dt\_pred == 0)]],  
 columns=['Fake Class', 'Test Data Labels', 'Random Forest', 'Support Vector Machine', 'Bernoulli Naive Bayes', 'Multi-layer Perceptron', 'Decision Tree'])

fgraph.plot(x='Fake Class', kind='bar', stacked=False, title='Comparison of Algorithms in Accuracy for the Fake Class')

<AxesSubplot:title={'center':'Comparison of Algorithms in Accuracy for the Fake Class'}, xlabel='Fake Class'>

png

png

lgraph = pd.DataFrame([[1, len(y\_test == 1), len(rfc\_pred == 1), len(svm\_pred == 1), len(bnb\_pred == 1), len(mlp\_pred == 1), len(dt\_pred == 1)]],  
 columns=['Legit Class', 'Test Data Labels', 'Random Forest', 'Support Vector Machine', 'Bernoulli Naive Bayes', 'Multi-layer Perceptron', 'Decision Tree'])

lgraph.plot(x='Legit Class', kind='bar', stacked=False, title='Comparison of Algorithms in Accuracy for the Legit Class')

<AxesSubplot:title={'center':'Comparison of Algorithms in Accuracy for the Legit Class'}, xlabel='Legit Class'>

png

png

rfc\_cm = confusion\_matrix(y\_test, rfc\_pred)  
svm\_cm = confusion\_matrix(y\_test, svm\_pred)  
bnb\_cm = confusion\_matrix(y\_test, bnb\_pred)  
mlp\_cm = confusion\_matrix(y\_test, mlp\_pred)  
dt\_cm = confusion\_matrix(y\_test, dt\_pred)

group\_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']  
group\_counts = ["{0:0.0f}".format(value) for value in rfc\_cm.flatten()]  
group\_percentages = ["{0:.2%}".format(value) for value in rfc\_cm.flatten()/np.sum(rfc\_cm)]  
labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names, group\_counts, group\_percentages)]  
labels = np.asarray(labels).reshape(2, 2)  
plt.title('Random Forest Classifier Confusion Matrix')  
sns.heatmap(rfc\_cm, annot=labels, fmt='', cmap='Blues')

<AxesSubplot:title={'center':'Random Forest Classifier Confusion Matrix'}>

png

png

group\_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']  
group\_counts = ["{0:0.0f}".format(value) for value in svm\_cm.flatten()]  
group\_percentages = ["{0:.2%}".format(value) for value in svm\_cm.flatten()/np.sum(svm\_cm)]  
labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names, group\_counts, group\_percentages)]  
labels = np.asarray(labels).reshape(2, 2)  
plt.title('Support Vector Classifier Confusion Matrix')  
sns.heatmap(rfc\_cm, annot=labels, fmt='', cmap='Blues')

<AxesSubplot:title={'center':'Support Vector Classifier Confusion Matrix'}>

png

png

group\_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']  
group\_counts = ["{0:0.0f}".format(value) for value in bnb\_cm.flatten()]  
group\_percentages = ["{0:.2%}".format(value) for value in bnb\_cm.flatten()/np.sum(bnb\_cm)]  
labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names, group\_counts, group\_percentages)]  
labels = np.asarray(labels).reshape(2, 2)  
plt.title('Bernoulli Naive Bayes Classifier Confusion Matrix')  
sns.heatmap(rfc\_cm, annot=labels, fmt='', cmap='Blues')

<AxesSubplot:title={'center':'Bernoulli Naive Bayes Classifier Confusion Matrix'}>

png

png

group\_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']  
group\_counts = ["{0:0.0f}".format(value) for value in mlp\_cm.flatten()]  
group\_percentages = ["{0:.2%}".format(value) for value in mlp\_cm.flatten()/np.sum(mlp\_cm)]  
labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names, group\_counts, group\_percentages)]  
labels = np.asarray(labels).reshape(2, 2)  
plt.title('Multi-layer Perceptron Confusion Matrix')  
sns.heatmap(rfc\_cm, annot=labels, fmt='', cmap='Blues')

<AxesSubplot:title={'center':'Multi-layer Perceptron Confusion Matrix'}>

png

png

group\_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']  
group\_counts = ["{0:0.0f}".format(value) for value in dt\_cm.flatten()]  
group\_percentages = ["{0:.2%}".format(value) for value in dt\_cm.flatten()/np.sum(dt\_cm)]  
labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names, group\_counts, group\_percentages)]  
labels = np.asarray(labels).reshape(2, 2)  
plt.title('Decision Tree Classifier Confusion Matrix')  
sns.heatmap(rfc\_cm, annot=labels, fmt='', cmap='Blues')

<AxesSubplot:title={'center':'Decision Tree Classifier Confusion Matrix'}>

png

png

print("Random Forest Classifier Accuracy: ", accuracy\_score(y\_test, rfc\_pred))  
print("Support Vector Classifier Accuracy: ", accuracy\_score(y\_test, svm\_pred))  
print("Bernoulli Naive Bayes Classifier Accuracy: ", accuracy\_score(y\_test, bnb\_pred))  
print("Multi-layer Perceptron Accuracy: ", accuracy\_score(y\_test, mlp\_pred))  
print("Decision Tree Classifier Accuracy: ", accuracy\_score(y\_test, dt\_pred))

Random Forest Classifier Accuracy: 0.9940898345153665  
Support Vector Classifier Accuracy: 0.8262411347517731  
Bernoulli Naive Bayes Classifier Accuracy: 0.966903073286052  
Multi-layer Perceptron Accuracy: 0.9929078014184397  
Decision Tree Classifier Accuracy: 0.9905437352245863

plt.bar('Random Forest', accuracy\_score(y\_test, rfc\_pred))  
plt.bar('SVM', accuracy\_score(y\_test, svm\_pred))  
plt.bar('BNB', accuracy\_score(y\_test, bnb\_pred))  
plt.bar('MLP', accuracy\_score(y\_test, mlp\_pred))  
plt.bar('DT', accuracy\_score(y\_test, dt\_pred))

<BarContainer object of 1 artists>

png

png

print("Random Forest Classifier F1-score: ", f1\_score(y\_test, rfc\_pred))  
print("Support Vector Classifier F1-score: ", f1\_score(y\_test, svm\_pred))  
print("Bernoulli Naive Bayes Classifier F1-score: ", f1\_score(y\_test, bnb\_pred))  
print("Multi-layer Perceptron F1-score: ", f1\_score(y\_test, mlp\_pred))  
print("Decision Tree Classifier F1-score: ", f1\_score(y\_test, dt\_pred))

Random Forest Classifier F1-score: 0.994535519125683  
Support Vector Classifier F1-score: 0.8093385214007782  
Bernoulli Naive Bayes Classifier F1-score: 0.9689578713968958  
Multi-layer Perceptron F1-score: 0.9934354485776805  
Decision Tree Classifier F1-score: 0.9912280701754386

plt.bar('Random Forest', f1\_score(y\_test, rfc\_pred))  
plt.bar('SVM', f1\_score(y\_test, svm\_pred))  
plt.bar('BNB', f1\_score(y\_test, bnb\_pred))  
plt.bar('MLP', f1\_score(y\_test, mlp\_pred))  
plt.bar('DT', f1\_score(y\_test, dt\_pred))

<BarContainer object of 1 artists>

png

png