Detect fake profiles in online social networks

TEAM: Sunil Sai Kumar Mada (20MCB1013)
Renuga keerthi (20MCB1017)
Bhargavinath Dornadula (20MCB1011)

About Dataset:

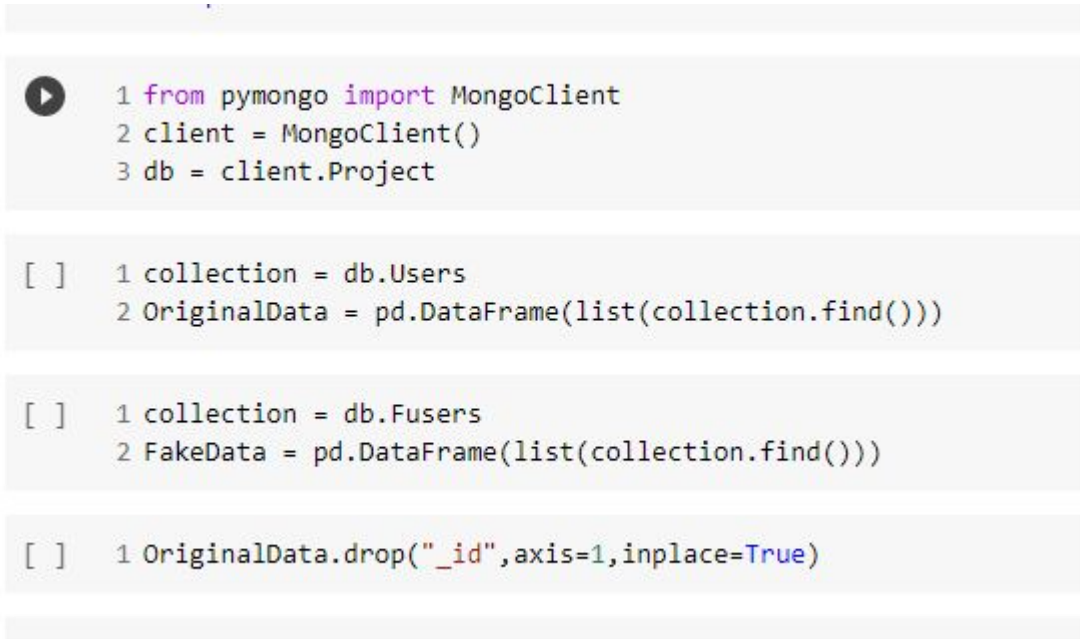
The dataset is been divided into 2 files : Users and Fake users

There are 1338 fake users and 1482 normal users with below mentioned attributes:

```
"id","name","screen_name","statuses_count","followers_count","friends_count","favourites_co
unt","listed_count","created_at","url","lang","time_zone","location","default_profile","def
ault_profile_image","geo_enabled","profile_image_url","profile_banner_url","profile_use_bac
kground_image","profile_background_image_url_https","profile_text_color","profile_image_url
_https","profile_sidebar_border_color","profile_background_tile","profile_sidebar_fill_colo
r","profile_background_image_url","profile_background_color","profile_link_color","utc_offs
et","protected","verified","description","updated","dataset"
```

Storing and Retrieving of Database

MongoDB has been used for storing of data



```
from pymongo import MongoClient
client = MongoClient()
db = client.Project
```

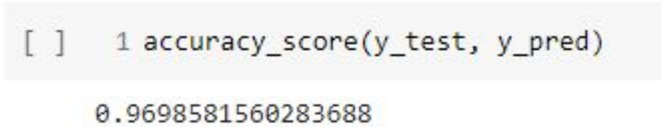
```
collection = db.Users
OriginalData = pd.DataFrame(list(collection.find()))
```

```
collection = db.Fusers
FakeData = pd.DataFrame(list(collection.find()))
```

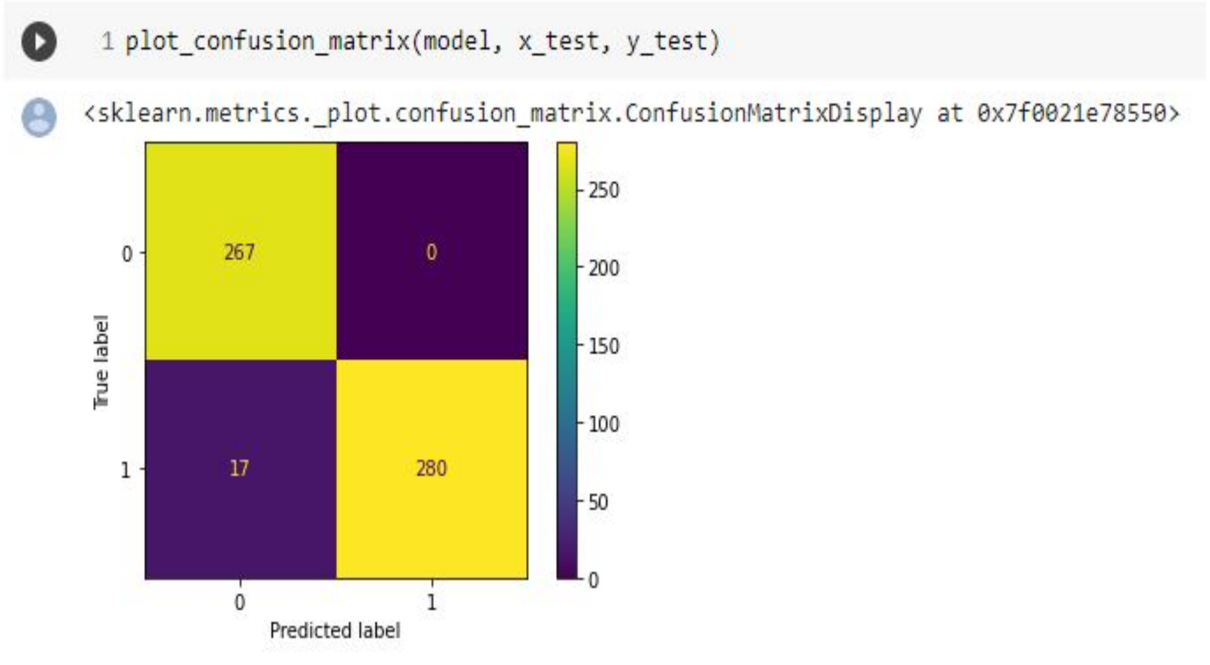
```
OriginalData.drop("_id",axis=1,inplace=True)
```

Models used:

1.Support Vector Machine(SVM):



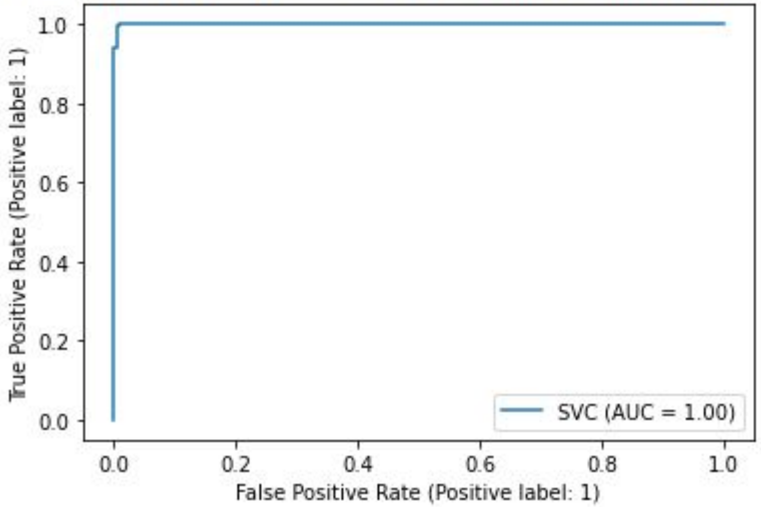
```
[ ]    accuracy_score(y_test, y_pred)

0.9698581560283688
```



```
plot_confusion_matrix(model, x_test, y_test)

<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f0021e78550>
```

```
plot_roc_curve(model, x_test, y_test)
```

```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7f00486e3100>
```



```
print(classification_report(y_test, y_pred, target_names=['Fake','Genuine']))
```

```
              precision    recall  f1-score   support

        Fake       0.94      1.00      0.97       267
     Genuine       1.00      0.94      0.97       297
```

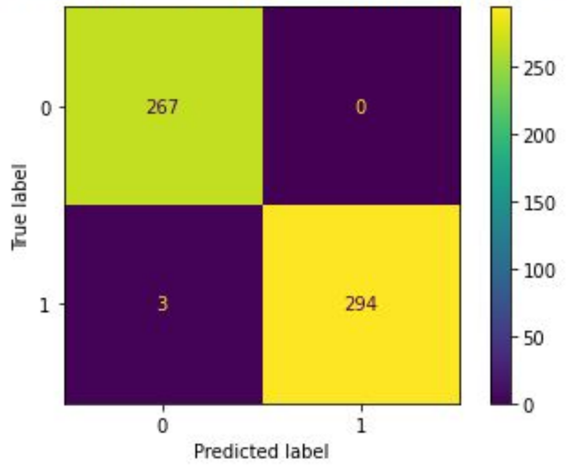
2. Logistic Regression:

```
[ ]   accuracy_score(y_test, prediction)
```

```
0.9946808510638298
```

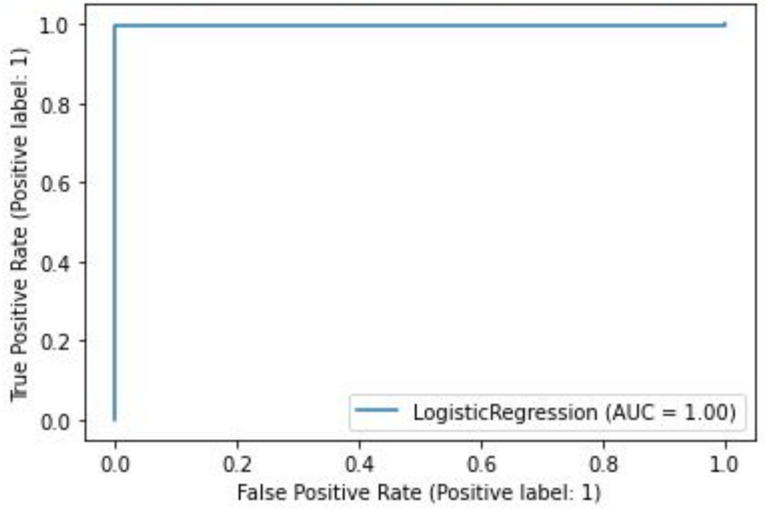
```
[ ]   plot_confusion_matrix(model, X_test, y_test)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f00218f3ee0>
```



```
plot_roc_curve(model, X_test, y_test)
```

```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7f0021901580>
```

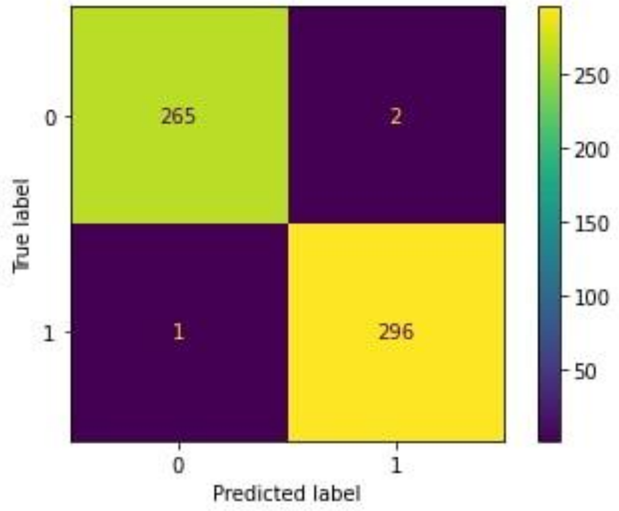


```
print(classification_report(y_test, prediction, target_names=['Fake','Genuine']))
```

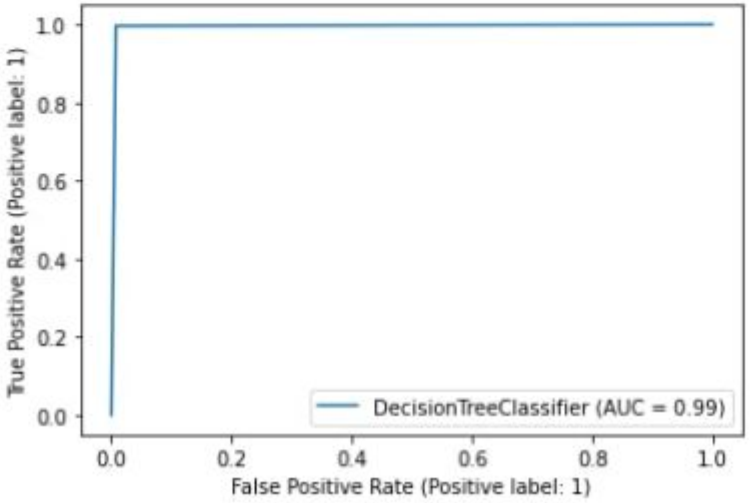
```
              precision    recall  f1-score   support

        Fake       0.99      1.00      0.99       267
     Genuine       1.00      0.99      0.99       297
```

3. Decision Tree:

```
In [36]: accuracy_score(y_test, y_pred)

Out[36]: 0.9946808510638298

In [37]: plot_confusion_matrix(classifier, X_test, y_test)

Out[37]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f6c720fbe80>
```



```
In [39]: plot_roc_curve(classifier, X_test, y_test)

Out[39]: <sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7f6c7215d580>
```



```
In [40]: print(classification_report(y_test, y_pred, target_names=['Fake','Genuine']))

              precision    recall  f1-score   support

        Fake       1.00      0.99      0.99       267
     Genuine       0.99      1.00      0.99       297
```