```
!pip install mlflow==1.20.2
```

```
Note: you may need to restart the kernel using dbutils.library.restartPython() to use updated packages.
Collecting mlflow==1.20.2
  Downloading mlflow-1.20.2-py3-none-any.whl (14.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.6/14.6 MB 39.4 MB/s eta 0:00:00
Requirement already satisfied: gitpython>=2.1.0 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (3.1.27)
Requirement already satisfied: databricks-cli>=0.8.7 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (0.18.
Requirement already satisfied: entrypoints in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (0.4)
Collecting querystring-parser
  Downloading querystring_parser-1.2.4-py2.py3-none-any.whl (7.9 kB)
Requirement already satisfied: packaging in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (22.0)
Collecting docker>=4.0.0
  Downloading docker-7.0.0-py3-none-any.whl (147 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 147.6/147.6 kB 19.8 MB/s eta 0:00:00
Requirement already satisfied: cloudpickle in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (2.0.0)
Requirement already satisfied: importlib-metadata!=4.7.0,>=3.7.0 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.
Requirement already satisfied: numpy in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (1.23.5)
Collecting alembic<=1.4.1
  Downloading alembic-1.4.1.tar.gz (1.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1 MB 41.0 MB/s eta 0:00:00
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: gunicorn in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (20.1.0)
Requirement already satisfied: pandas in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (1.5.3)
Requirement already satisfied: sqlparse>=0.3.1 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (0.4.2)
Requirement already satisfied: sqlalchemy in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (1.4.39)
Requirement already satisfied: requests>=2.17.3 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (2.28.1)
Requirement already satisfied: Flask in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (2.2.5)
Requirement already satisfied: protobuf>=3.7.0 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (4.24.0)
Requirement already satisfied: click>=7.0 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (8.0.4)
Requirement already satisfied: pytz in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (2022.7)
Collecting prometheus-flask-exporter
  Downloading prometheus_flask_exporter-0.23.0-py3-none-any.whl (18 kB)
Requirement already satisfied: pyyaml>=5.1 in /databricks/python3/lib/python3.10/site-packages (from mlflow==1.20.2) (6.0)
Requirement already satisfied: Mako in /databricks/python3/lib/python3.10/site-packages (from alembic<=1.4.1->mlflow==1.20.2) (1.2.0)
Requirement already satisfied: python-editor>=0.3 in /databricks/python3/lib/python3.10/site-packages (from alembic<=1.4.1->mlflow==1
Requirement already satisfied: python-dateutil in /databricks/python3/lib/python3.10/site-packages (from alembic<=1.4.1->mlflow==1.20
Requirement already satisfied: tabulate>=0.7.7 in /databricks/python3/lib/python3.10/site-packages (from databricks-cli>=0.8.7->mlflo
Requirement already satisfied: urllib3<3,>=1.26.7 in /databricks/python3/lib/python3.10/site-packages (from databricks-cli>=0.8.7->ml
Requirement already satisfied: six>=1.10.0 in /usr/lib/python3/dist-packages (from databricks-cli>=0.8.7->mlflow==1.20.2) (1.16.0)
Requirement already satisfied: pyjwt>=1.7.0 in /usr/lib/python3/dist-packages (from databricks-cli>=0.8.7->mlflow==1.20.2) (2.3.0)
Requirement already satisfied: oauthlib>=3.1.0 in /usr/lib/python3/dist-packages (from databricks-cli>=0.8.7->mlflow==1.20.2) (3.2.0)
Requirement already satisfied: gitdb<5,>=4.0.1 in /databricks/python3/lib/python3.10/site-packages (from gitpython>=2.1.0->mlflow==1.
Requirement already satisfied: zipp>=0.5 in /databricks/python3/lib/python3.10/site-packages (from importlib-metadata!=4.7.0,>=3.7.0-
Requirement already satisfied: certifi>=2017.4.17 in /databricks/python3/lib/python3.10/site-packages (from requests>=2.17.3->mlflow=
Requirement already satisfied: idna<4,>=2.5 in /databricks/python3/lib/python3.10/site-packages (from requests>=2.17.3->mlflow==1.20.
Requirement already satisfied: charset-normalizer<3,>=2 in /databricks/python3/lib/python3.10/site-packages (from requests>=2.17.3->m
Requirement already satisfied: greenlet!=0.4.17 in /databricks/python3/lib/python3.10/site-packages (from sqlalchemy->mlflow==1.20.2)
Requirement already satisfied: itsdangerous>=2.0 in /databricks/python3/lib/python3.10/site-packages (from Flask->mlflow==1.20.2) (2.
Requirement already satisfied: Jinja2>=3.0 in /databricks/python3/lib/python3.10/site-packages (from Flask->mlflow==1.20.2) (3.1.2)
Requirement already satisfied: Werkzeug>=2.2.2 in /databricks/python3/lib/python3.10/site-packages (from Flask->mlflow==1.20.2) (2.2.
Requirement already satisfied: setuptools>=3.0 in /databricks/python3/lib/python3.10/site-packages (from gunicorn->mlflow==1.20.2) (6
Requirement already satisfied: prometheus-client in /databricks/python3/lib/python3.10/site-packages (from prometheus-flask-exporter-
Requirement already satisfied: smmap<6,>=3.0.1 in /databricks/python3/lib/python3.10/site-packages (from gitdb<5,>=4.0.1->gitpython>=
Requirement already satisfied: MarkupSafe>=2.0 in /databricks/python3/lib/python3.10/site-packages (from Jinja2>=3.0->Flask->mlflow==
Building wheels for collected packages: alembic
  Building wheel for alembic (setup.py): started
  Building wheel for alembic (setup.py): finished with status 'done'
```

```
import numpy as np
import pandas as pd
import matplotlib #
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn #
#import spark
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import roc_auc_score, plot_roc_curve,confusion_matrix
from sklearn.model_selection import KFold
import mlflow
```

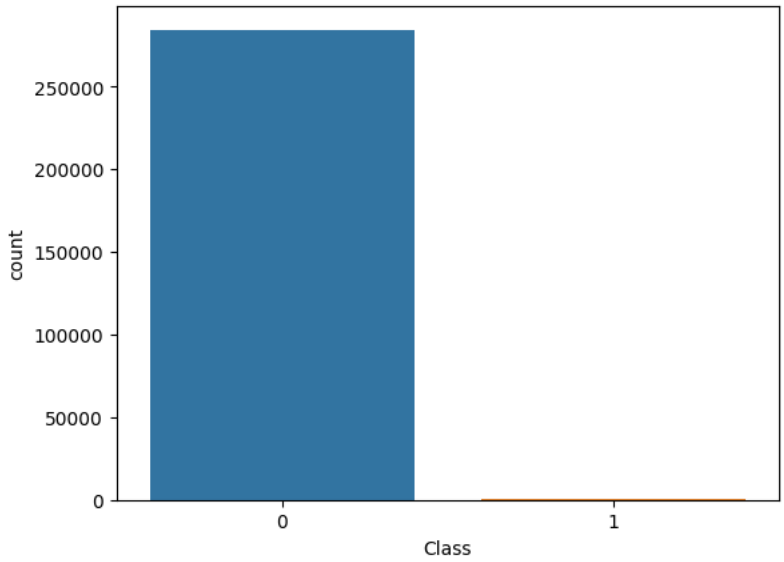
```
# File location and type
file_location = "/FileStore/shared_uploads/imad.elachiri@um6p.ma/creditcard.csv"
file_type = "csv"

# CSV options
infer_schema = "true"
first_row_is_header = "true"
delimiter = ","

# The applied options are for CSV files. For other file types, these will be ignored.
dataframe = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location)

df=dataframe.toPandas()
```

```
df = df.drop("Time", axis=1) # cette colonne n'apporte rien à notre analyse
df.head()
```

	V1	V2	V3	V4	V5	V6	V7	V8	
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817

```
sns.countplot(x='Class',data=df)
```

```
<Axes: xlabel='Class', ylabel='count'>
```




```
normal = df[df.Class == 0].sample(frac=0.5,random_state=2020).reset_index(drop=True)
anomaly = df[df.Class == 1]
```

```
print(f"Normal: {normal.shape}")
print(f"Anomaly: {anomaly.shape}")
```

```
Normal: (142158, 30)
Anomaly: (492, 30)
```

```
normal_train, normal_test = train_test_split(normal,test_size = 0.2, random_state = 2020)
anomaly_train, anomaly_test = train_test_split(anomaly, test_size = 0.2, random_state = 2020)
normal_train, normal_validate = train_test_split(normal_train,test_size = 0.25, random_state = 2020)
anomaly_train, anomaly_validate = train_test_split(anomaly_train, test_size = 0.25, random_state = 2020)
```

```
x_train = pd.concat((normal_train, anomaly_train))
x_test = pd.concat((normal_test, anomaly_test))
x_validate = pd.concat((normal_validate, anomaly_validate))


y_train = np.array(x_train["Class"])
y_test = np.array(x_test["Class"])
y_validate = np.array(x_validate["Class"])


x_train = x_train.drop("Class", axis=1)
x_test = x_test.drop("Class", axis=1)
x_validate = x_validate.drop("Class", axis=1)


print("Training sets:\nx_train: {} \ny_train:{}".format(x_train.shape, y_train.shape))
print("\nTesting sets:\nx_test: {} \ny_test:{}".format(x_test.shape, y_test.shape))
print("\nValidation sets:\nx_validate: {} \ny_validate: {}".format(x_validate.shape, y_validate.shape))
```

```
Training sets:
x_train: (85588, 29)
y_train:(85588,)

Testing sets:
x_test: (28531, 29)
y_test:(28531,)

Validation sets:
x_validate: (28531, 29)
y_validate: (28531,)
```

```
scaler = StandardScaler()
scaler.fit(pd.concat((normal, anomaly)).drop("Class", axis=1))
```

```
▾ StandardScaler
StandardScaler()
```

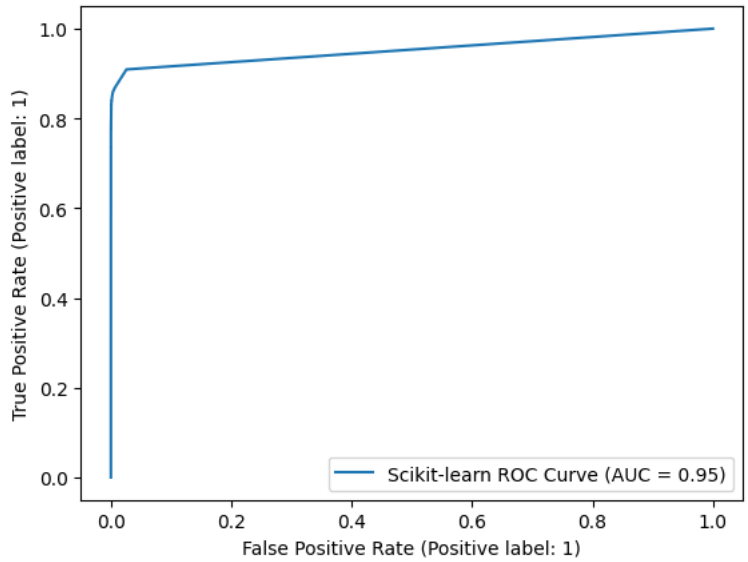
```
x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)
x_validate = scaler.transform(x_validate)


def train(sk_model, x_train, y_train):
    sk_model = sk_model.fit(x_train, y_train)
    train_acc = sk_model.score(x_train, y_train)
    mlflow.log_metric("train_acc", train_acc)
    print(f"Train Accuracy: {train_acc:.3%}")


def evaluate(sk_model, x_test, y_test):
    eval_acc = sk_model.score(x_test, y_test)
    preds = sk_model.predict(x_test)
    auc_score = roc_auc_score(y_test, preds)
    mlflow.log_metric("eval_acc", eval_acc) # nous avons fait la même chose pour l'accuracy de test (evaluation)
    mlflow.log_metric("auc_score", auc_score)# aussi pour l'AUC score
    print(f"Auc Score: {auc_score:.3%}")
    print(f"Eval Accuracy: {eval_acc:.3%}")
    roc_plot = plot_roc_curve(sk_model, x_test, y_test,name='Scikit-learn ROC Curve')
    plt.savefig("sklearn_roc_plot.png")
    plt.show()
    plt.clf()
    conf_matrix = confusion_matrix(y_test, preds)
    ax = sns.heatmap(conf_matrix, annot=True,fmt='g')
    ax.invert_xaxis()
    ax.invert_yaxis()
    plt.ylabel('Actual')
    plt.xlabel('Predicted')
    plt.title("Confusion Matrix")
    plt.savefig("sklearn_conf_matrix.png")
    # nous avons dit à MLflow de sauvegarder ces deux figure afin de les consulter à chaque
    # exécution d'une façon organisée comme nous allons le voir plus loin dans cet atelier
    mlflow.log_artifact("sklearn_roc_plot.png")
    mlflow.log_artifact("sklearn_conf_matrix.png")
```

```
#to configure your DataBricks CLI
token = dbutils.notebook.entry_point.getDbutils().notebook().getContext().apiToken().get()
dbutils.fs.put("file:///root/.databrickscfg","[DEFAULT]\nhost=https://community.cloud.databricks.com\ntoken ="+token,overwrite=True)
```

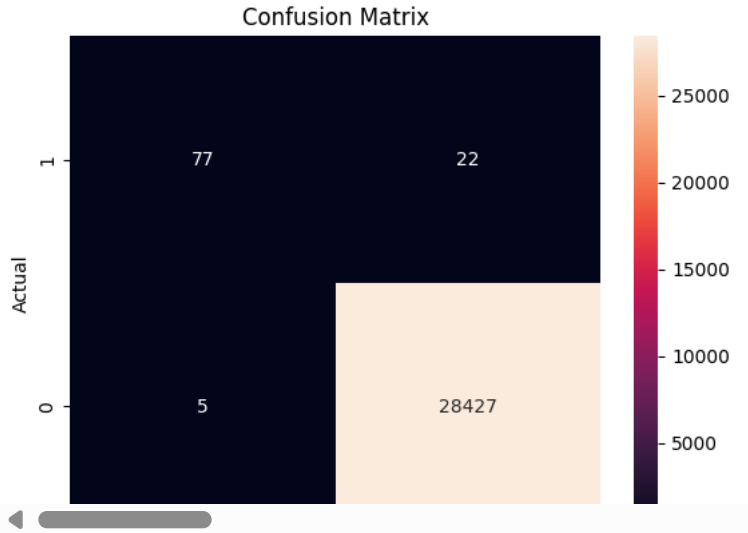
```
Wrote 97 bytes.
True
```

```
sk_model = RandomForestClassifier(n_estimators=50)
mlflow.set_experiment("/Users/imad.elachiri@um6p.ma/experiment-DataBricks1")
with mlflow.start_run():
    train(sk_model, x_train, y_train)
    evaluate(sk_model, x_test, y_test)
    mlflow.sklearn.log_model(sk_model, "RF_model")
    print("Model run: ", mlflow.active_run().info.run_uuid)
mlflow.end_run()
```

```
Train Accuracy: 99.996%
Auc Score: 88.880%
Eval Accuracy: 99.905%
/databricks/python/lib/python3.10/site-packages/sklearn/utils/deprecation.py:87: Futu
  warnings.warn(msg, category=FutureWarning)
```

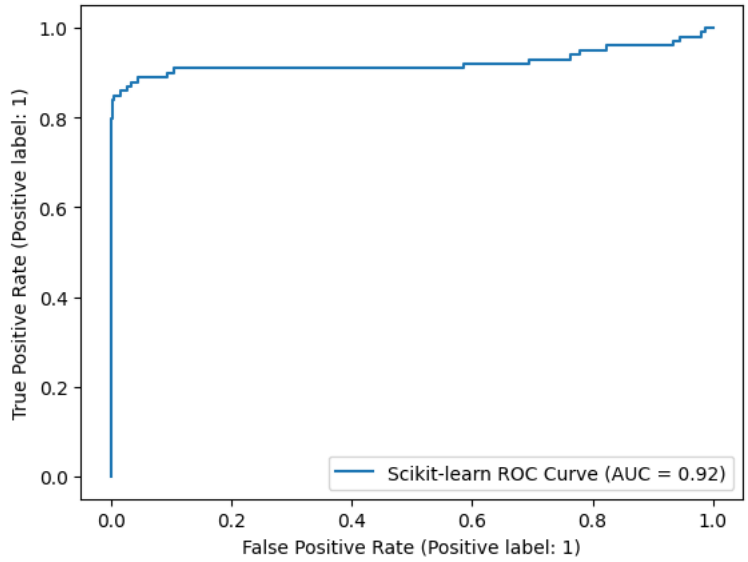



```
2024/01/08 02:20:15 WARNING mlflow.models.model: Model logged without a signature. Si
2024/01/08 02:20:15 WARNING mlflow.utils.environment: Encountered an unexpected error
Uploading artifacts:   0%|          | 0/5 [00:00<?, ?it/s]
Model run:  fb6945a5b3ed41698d263385740d50ee
```

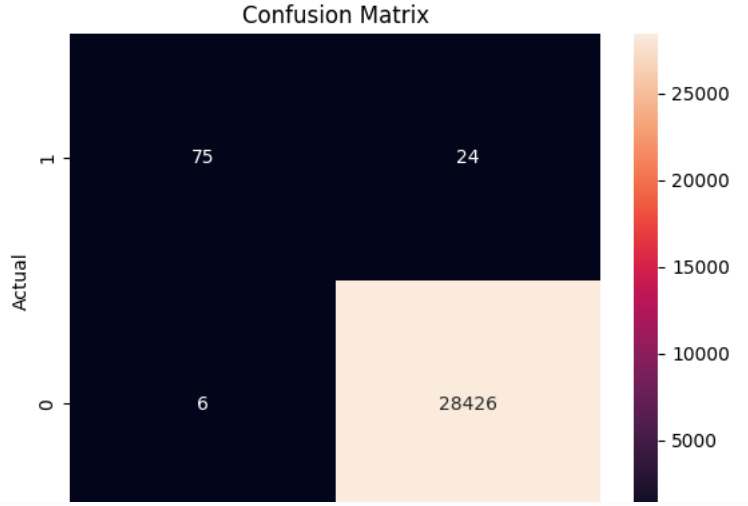


```
sk_model = SVC(C=0.9, kernel="poly")
mlflow.set_experiment("/Users/imad.elachiri@um6p.ma/experiment-DataBricks1")
with mlflow.start_run():
    train(sk_model, x_train, y_train)
    evaluate(sk_model, x_test, y_test)
    mlflow.sklearn.log_model(sk_model, "MN_NB_model")
    print("Model run: ", mlflow.active_run().info.run_uuid)
mlflow.end_run()
```

```
Train Accuracy: 99.959%
Auc Score: 87.868%
Eval Accuracy: 99.895%
/databricks/python/lib/python3.10/site-packages/sklearn/utils/deprecation.py:87: Futu
  warnings.warn(msg, category=FutureWarning)
```




```
2024/01/08 02:15:33 WARNING mlflow.models.model: Model logged without a signature. Si
2024/01/08 02:15:33 WARNING mlflow.utils.environment: Encountered an unexpected error
Uploading artifacts:   0%|          | 0/5 [00:00<?, ?it/s]
Model run:  487a5717852843978fabf189c899ab06
```



```
# define evaluation

# define search space
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import loguniform
# summarize result
#print('Best Score: %s' % result.best_score_)
#print('Best Hyperparameters: %s' % result.best_params_)

model = LogisticRegression()
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
space = dict()
space['solver'] = ['newton-cg', 'lbfgs', 'liblinear']
space['penalty'] = ['none', 'l1', 'l2', 'elasticnet']
space['C'] = loguniform(1e-5, 100)
```

```
all_runs = mlflow.search_runs(max_results=10)  # Note : This is pandas dataframe
display(all_runs)
```

run_id	experiment_id	status	artifact_uri
fb6945a5b3ed41698d263385740d50ee	2041523735748677	FINISHED	dbfs:/databricks/mlflow-tracking/204152373574867
119f6561b26d4a74b880e1d9909c6506	2041523735748677	FINISHED	dbfs:/databricks/mlflow-tracking/204152373574867
487a5717852843978fabf189c899ab06	2041523735748677	FINISHED	dbfs:/databricks/mlflow-

TAF:

1. réexécutez la cellule 18 en utilisant d'autres modèle ML de SKlearn
2. comparez vos résultats en utilisant l'outil de comparaison de MLFlow

```
print()
```