HW6 - David Euijoon Kim - Machine Learning Predictions
https://github.com/dk-davidekim/Google-Cloud-Computing.git

1. Python (BaseLine)

```python
import os
import pandas as pd
import sqlalchemy
from dotenv import load_dotenv
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
from google.cloud.sql.connector import Connector
from google.cloud import logging

load_dotenv()

class DatabaseConnector:
    def __init__(self):
        db_connection_string = os.getenv('DB_CONNECTION_STRING')
        db_user = os.getenv('DB_USER')
        db_password = os.getenv('DB_PASSWORD')
        db_database = os.getenv('DB_DATABASE')

        self.connector = Connector()

        def getconn():
            return self.connector.connect(
                db_connection_string,
                "pymysql",
                user=db_user,
                password=db_password,
                db=db_database
            )

        self.pool = sqlalchemy.create_engine(
            "mysql+pymysql://",
            creator=getconn
        )

class Logger:
    def __init__(self):
        project_id = os.getenv('GOOGLE_CLOUD_PROJECT_ID')
        logger_name = 'hw6'
        self.logging_client = logging.Client(project=project_id)
        self.logger = self.logging_client.logger(logger_name)

    def log(self, message):
        print(message)  # Print message to stdout as well
        self.logger.log_text(message)

```

2. Python (Predict Country - RandomForestClassifier)

```python
class Prediction:
    def __init__(self, db_interface):
        self.logger = Logger()
        self.db_interface = db_interface
        self.c_classifier = None
        self.i_classifier = None
        self.i_label_encoder = LabelEncoder()

    def predict_country(self):
        with self.db_interface.pool.connect() as conn:
            fetch_query = sqlalchemy.sql.text("SELECT client_ip, country FROM Clients")
            data = pd.read_sql(fetch_query, conn)

            data['binary_ip'] = data['client_ip'].apply(
                lambda ip: int(''.join([f'{int(byte):08b}' for byte in ip.split('.')]), 2)
            )

            features = data[['binary_ip']]
            labels = data['country']

            feature_train, feature_test, label_train, label_test = train_test_split(
                features, labels, test_size=0.3, random_state=0
            )

            model = RandomForestClassifier()
            model.fit(feature_train, label_train)

            predictions = model.predict(feature_test)
            accuracy = accuracy_score(label_test, predictions)
            self.logger.log(f'Accuracy of IP Classifier: {accuracy * 100:.2f}%')

            self.c_classifier = model
```

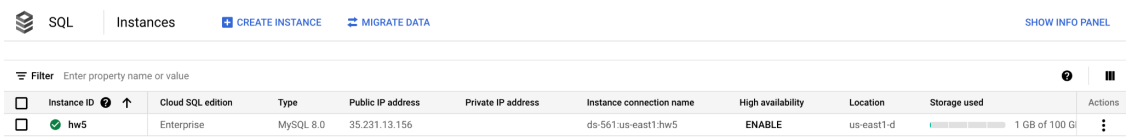
3. Python (Predict Income - Neural Network)

```python
    def predict_income(self):
        with self.db_interface.pool.connect() as conn:
            fetch_query = sqlalchemy.sql.text("SELECT gender, age, country, is_banned, income FROM Clients")
            data = pd.read_sql(fetch_query, conn)

            y = self.i_label_encoder.fit_transform(data['income'])
            data = pd.get_dummies(data, columns=['gender', 'age', 'country', 'is_banned'])
            X = data.drop(['income'], axis=1)

            X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

            mlp = MLPClassifier(hidden_layer_sizes=(50,), activation='relu', solver='adam', alpha=0.0001, learning_rate='adaptive', max_iter=100)
            mlp.fit(X_train, y_train)

            predictions = mlp.predict(X_test)
            accuracy = accuracy_score(y_test, predictions)
            self.logger.log(f'Accuracy of Income Classifier: {accuracy * 100:.2f}%')

            self.i_classifier = mlp

if __name__ == "__main__":
    db_connector = DatabaseConnector()
    db_interface = type('DBInterface', (object,), {'pool': db_connector.pool})
    prediction = Prediction(db_interface)
    prediction.predict_country()
    prediction.predict_income()
```

4. Python (Predict Income - Random Forest)

```python
    def predict_income(self):
        with self.db_interface.pool.connect() as conn:
            fetch_query = sqlalchemy.sql.text("SELECT gender, age, country, is_banned, income FROM Clients")
            data = pd.read_sql(fetch_query, conn)

            # Encode labels
            y = self.i_label_encoder.fit_transform(data['income'])
            data = pd.get_dummies(data, columns=['gender', 'age', 'country', 'is_banned'])
            X = data.drop(['income'], axis=1)

            # Split dataset
            X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

            model = RandomForestClassifier()
            model.fit(X_train, y_train)

            predictions = model.predict(X_test)
            accuracy = accuracy_score(y_test, predictions)
            self.logger.log(f'Accuracy of IP Classifier: {accuracy * 100:.2f}%')

            self.c_classifier = model

if __name__ == "__main__":
    db_connector = DatabaseConnector()
    db_interface = type('DBInterface', (object,), {'pool': db_connector.pool})
    prediction = Prediction(db_interface)
    prediction.predict_country()
    prediction.predict_income()

```

5. Make sure SQL Database is Turned On



6. Create a VM

7. Send Files to VM

gcloud compute scp
/Users/davidekim/Desktop/DataScience/BU/DS561/ds561-davidekim-U66545284/hw6/hw6_NN.py hw6:~/hw6_NN.py --zone us-east1-b

gcloud compute scp
/Users/davidekim/Desktop/DataScience/BU/DS561/ds561-davidekim-U66545284/hw6/.env hw6:~/.env --zone us-east1-b

gcloud compute scp
/Users/davidekim/Desktop/DataScience/BU/DS561/ds561-davidekim-U66545284/hw6/hw6_RF.py hw6:~/hw6_RF.py --zone us-east1-b

8. Download Libraries

sudo apt-get update
sudo apt-get install python3 python3-pip

pip install pandas sqlalchemy python-dotenv scikit-learn cloud-sql-python-connector google-cloud-logging pymysql

9. Run Files

python3 hw6_NN.py

```
dk98@hw6:/home/davidekim$ python3 hw6_NN.py
Accuracy of IP Classifier: 99.97%
/home/dk98/.local/lib/python3.9/site-packages/sklearn/neural_network/_multilayer_perceptron.py:691: Convergence
Warning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
Accuracy of Income Classifier: 12.62%
```

python3 hw6_RF.py

```
dk98@hw6:/home/davidekim$ python3 hw6_RF.py
Accuracy of IP Classifier: 99.97%
Accuracy of IP Classifier: 12.38%
```

*Selecting features and tuning the model parameters did not help increase the accuracy because the client information are randomly mixed and added to the data from httpclient.py.

*Random Forest is a ensemble machine learning model where multitude of decision trees are operated.
*Neural network is a deep learning model that uses interconnected neurons in a layered structure.

END.