# EDP Guided Project 3

In this project, you will create a solution to help predict if a Star Wars character will join the Empire or the Resistance based on their home planet and unit type. You will start by reviewing a Python script that will help you generate sample data to train a model. Once the model has been built, you will use it to predict a 10 million record data set.

Good luck and may The Force be with you.

# Part 1: Review a Python script to generate data

Review the Python script named **generate\_data.py** (see Guided Project zip for file). The script generates data for 1000 Star Wars characters. Each record will have a:

* timestamp
* unit\_id (sequential integer)
* unit\_type (random selection from sequence below)

unit\_type=["stormtrooper", "tie\_fighter", "at-st", "x-wing", "resistance\_soldier", "at-at", "tie\_silencer", "unknown"]

* empire\_or\_resistance (random selection of either empire or resistance)
* location\_x
* location\_y
* destination\_x
* destination\_y
* home\_world (read data from a file named **home\_worlds.json,** see Guided Project zip for file, and select a random homeworld)

The generated data is saved to a file named **troop\_movements.csv**. The x and y co-ordinates simulate troops moving from one position to another in battle on an imaginary grid.

Here’s an example of what the data should look like when read into a data frame.

A screenshot of a computer

Description automatically generated with low confidence

Since we are generating random data, there will be “bad” Empire characters ending up as Resistance and vice versa.

# Part 2: Build a prediction model

Read the generated data from the csv file into a Pandas data frame and explore the data.

**NOTE: All images below are just examples. Your numbers will be different.**

* Create grouped data showing counts of empire vs resistance.  
  A black text on a white background

  Description automatically generated with low confidence
* Create grouped data showing counts of characters by homeworld.  
  A picture containing text, font, screenshot, number

  Description automatically generated
* Created grouped data showing counts of characters by unit\_type.  
  A picture containing text, font, screenshot, receipt

  Description automatically generated
* Engineer a new feature called is\_resistance with a True or False value based on empire\_or\_resiatance.  
  A picture containing text, number, line, font

  Description automatically generated
* Create a bar plot using Seaborn showing Empire vs Resistance distribution.  
  A picture containing text, screenshot, diagram

  Description automatically generated
* Create a prediction model using **sklearn.tree.DecisionTreeClassifier** that predicts if a character is joining either the Empire or the Resistance based on their homeworld and unit\_type.
* Convert categorical features to numeric using **pd.get\_dummies**.
* Create a bar plot that shows feature importance.
* Example code to get feature importance:

# Get feature importances

importances = model.feature\_importances\_

# Create a DataFrame to hold the feature importances

feature\_importances = pd.DataFrame({'Feature': X\_encoded.columns, 'Importance': importances})

A picture containing text, screenshot, diagram, plot

Description automatically generated

* Save to model as a pickle file named **trained\_model.pkl**.

# Part 3: Use the trained model with “real” data

Load data from **troop\_movements10m.csv** (see Guided Project zip for file). This file contains 10 million records to be predicted.

This data must be cleaned up a bit before it can be used:

* Some unit\_type records have a value of invalid\_unit. Replace that with **unknown**.
* Some location\_x and location\_Y values are missing. Use the **ffill** method to fill.
* Save the clean data into a Parquet file named troop\_movements10m.parquet.
  + You need to install pyarrow and fastparquet to support saving to a Parquet file.

pip install pyarrow

pip install fastparquet

Load the pickled model and load the data from the Parquet file into a data frame. Run the data through the model.

Add the predicted values to the data frame. The finished result should look like this:

**NOTE: Your values will be different.**

A picture containing text, font, number, line

Description automatically generated