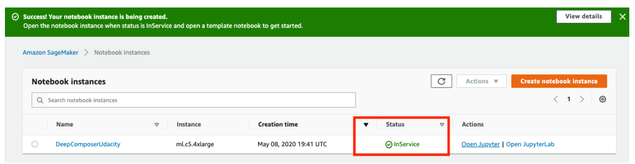
When the status reads "InService" you can open the Jupyter notebook.



Open the Notebook

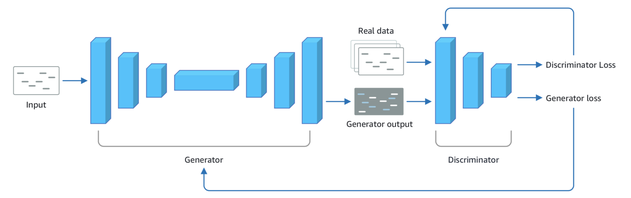
1. Click **Open Jupyter**.
2. When the notebook opens, click on **Lab 2**.
3. When the lab opens click on **GAN.ipynb**.

Review: Generative Adversarial Networks (GANs).

GANs consist of two networks constantly competing with each other:

- **Generator network** that tries to generate data based on the data it was trained on.
- **Discriminator network** that is trained to differentiate between real data and data which is created by the generator.

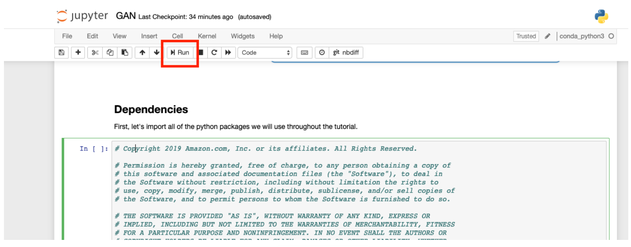
Note: The demo often refers to the **discriminator** as the **critic**. The two terms can be used interchangeably.



Set Up the Project

1. Run the first **Dependencies** cell to install the required packages
2. Run the second **Dependencies** cell to import the dependencies
3. Run the **Configuration** cell to define the configuration variables

Note: While executing the cell that installs dependency packages, you may see warning messages indicating that later versions of conda are available for certain packages. It is completely OK to ignore this message. It should not affect the execution of this notebook.



Click **Run** or `Shift-Enter` in the cell

Good Coding Practices

- Do not hard-code configuration variables
- Move configuration variables to a separate `config` file
- Use code comments to allow for easy code collaboration

Data Preparation

The next section of the notebook is where we'll prepare the data so it can train the generator network.

Why Do We Need to Prepare Data?

Data often comes from many places (like a website, IoT sensors, a hard drive, or physical paper) and it's usually not clean or in the same format. Before you can better understand your data, you need to

make sure it's in the right format to be analyzed. Thankfully, there are library packages that can help! One such library is called NumPy, which was imported into our notebook.

Piano Roll Format

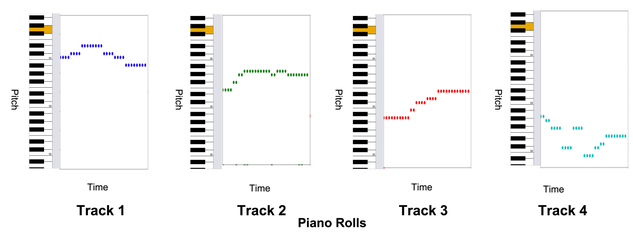
The data we are preparing today is music and it comes formatted in what's called a "piano roll". Think of a piano roll as a 2D image where the X-axis represents time and the Y-axis represents the pitch value. Using music as images allows us to leverage existing techniques within the computer vision domain.

Our data is stored as a `NumPy` Array, or grid of values. Our dataset comprises 229 samples of 4 tracks (all tracks are piano). Each sample is a 32 time-step snippet of a song, so our dataset has a shape of:

```
(num_samples, time_steps, pitch_range, tracks)
```

or

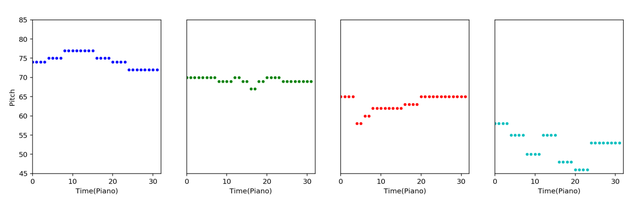
```
(229, 32, 128, 4)
```



Each Piano Roll Represents A Separate Piano Track in the Song

Load and View the Dataset

4. Run the next cell to play a song from the dataset.
5. Run the next cell to load the dataset as a `nympy` array and output the shape of the data to confirm that it matches the `(229, 32, 128, 4)` shape we are expecting
6. Run the next cell to see a graphical representation of the data.



Graphical Representation of Model Data

Create a Tensorflow Dataset

Much like there are different libraries to help with cleaning and formatting data, there are also different frameworks. Some frameworks are better suited for particular kinds of machine learning workloads and for this deep learning use case, we're going to use a Tensorflow framework with a Keras library.

We'll use the dataset object to feed batches of data into our model.

7. Run the first Load Data cell to set parameters.
8. Run the second Load Data cell to prepare the data.

NEXT