GROUP 19

PROJECT-

DOPPELGANGER

Table of Contents

1. EXECUTIVE SUMMARY

WE ARE ALL BECOMING INCREASINGLY AWARE OF THE VALUE OF OUR DATA, and the desire to share it without the concept of a value exchange is dwindling. A

true and widely accepted model for the value exchange has yet to be developed, and as a result the ability for organisations to share data is slowing down data innovation. From an organisation perspective, regulations like GDPR and an increased desire for privacy among consumers are driving this cautionary approach when it comes to data. As a result, these organisations are keen to embrace technological advances that mean they can share data and derive insights whilst maintaining compliance with the demands of both consumers and regulators. There are a number of ways in which this problem is being approached, but the one that I want to discuss in this penultimate article of the data series is **Synthetic Data**.

As synthetic data is anonymous and exempt from data protection regulations, this opens up a whole range of opportunities for otherwise locked-up data, resulting in faster innovation, less risk and lower costs. This article covers what it is, how it's generated and the potential applications.

2. BUSINESS PROBLEM BACKGROUND

2.1. Why Synthetic data?

Data scientists all around the world are craving for data. The desire to train and deploy cutting-edge machine learning algorithms like neural networks pushes the need for more data to the next level. This quickly poses a problem when new data collection is tedious, costly or simply impossible. Synthetic data gained more and more popularity as of lately, since it promises to fulfil the need for large amounts of data. The possibility to just create some "fake" data, that for instance can subsequently be used as training data for machine learning models, sounds very promising. However, one should not fall into the trap of thinking that synthetic data is the holy grail of data science that solves all problems. In this project, we will illustrate the usefulness of synthetic data as well as discuss the common pitfalls that may arise when synthetic data is used for real use cases.

2.2. Data protection in the EU

The data protection package adopted in May 2016 aims at making Europe fit for the digital age. More than 90% of Europeans say they want the same data protection rights across the EU and regardless of where their data is processed.

2.2.1. The General Data Protection Regulation (GDPR)

Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data. This text includes the corrigendum published in the OJEU of 23 May 2018.

The regulation is an essential step to strengthen individuals' fundamental rights in the digital age and facilitate business by clarifying rules for companies and public bodies in the digital single market. A single law will also do away with the current fragmentation in different national systems and unnecessary administrative burdens.

The regulation entered into force on 24 May 2016 and applies since 25 May 2018. More information for companies and individuals.

Information about the incorporation of the General Data Protection Regulation (GDPR) into the EEA Agreement.

National data protection authorities

EU countries have set up national bodies responsible for protecting personal data in accordance with Article 8(3) of the Charter of Fundamental Rights of the EU.

European Data Protection Board

The European Data Protection Board (EDPB) is an independent European body which shall ensure the consistent application of data protection rules throughout the European Union. The EDPB has been established by the General Data Protection Regulation (GDPR).

3. ABOUT PROJECT

3.1. What is synthetic data?

Synthetic data generation describes a method of producing artificial datapoints from a real dataset. The new data is supposed to mimic the original data such that the two datasets cannot be distinguished from one another, not even by human domain experts or computer algorithms. Having more data with similar properties to the original can be useful in a variety of ways. For example, machine learning models often improve in performance, the more training data is fed to them. Using synthetic data, more and complementary data can be created that eventually might improve a model.

3.2. How can synthetic data be generated?

There are numerous ways to create synthetic data, each one with their own advantages and limitations. Often neural networks or Bayesian networks are utilised in order to generate new data. The following sections provide an overview of the most common tools.

3.2.1. Neural Networks

Numerous methods for generating synthetic data utilise neural networks, for example *variational autoencoders* (VAE) that learn patterns in data by utilizing encoding and decoding techniques or *autoregressive models* that are used to generate synthetic images. Probably the most popular method for producing synthetic data today are *Generative Adversarial Networks* (or GANs).

3.3. Applications of synthetic data

Whenever privacy concerns are an issue such as in the financial and healthcare industries or an enormous data set is required to train machine learning algorithms, synthetic data sets can propel progress. Here are just a few applications of synthetic data:

- Synthetic data with record-level data can be used from healthcare organizations to inform care protocols while protecting patient confidentiality. Simulated X-rays are combined with actual X-rays to train AI algorithms to identify conditions.
- Fraudulent activity detection systems can be tested and trained without exposing personal financial records.
- DevOps teams use synthetic data to test software and ensure quality.
- Machine learning algorithms are often trained with synthetic data.
- Waymo tested its autonomous vehicles by driving 8 million miles on real roads plus another 5 billion on simulated roadways. Other automakers are using video games such as Grand Theft Auto to aid its self-driving technology.

While synthetic data isn't fool proof, it is an important tool to augment machine learning algorithms when real data is too expensive to collect, inaccessible due to privacy concerns or incomplete.

3.4. Disadvantages of synthetic data

It can be challenging to create high-quality synthetic data especially if the system is complex. It's important that the generative model creating the synthetic data is excellent or the data it generates will be affected. If synthetic data isn't nearly identical to a real-world data set, it can compromise the quality of decision-making that is being done based on the data.

Even if synthetic data is really good, it is still a replica of specific properties of a real data set. A model looks for trends to replicate, so some of the random behaviors might be missed.

3.5. Advantages of synthetic data

Huge data sets are what powers deep learning machines and artificial intelligence algorithms that are expected to help solve very challenging issues. Companies such as Google, Facebook and Amazon have had a competitive advantage due to the amount of data they create daily as part of their business. Synthetic data allows organizations of every size and resource levels the possibility to also capitalize on learning that is powered by deep data sets which ultimately can democratize mahine learning.

Creating synthetic data is more efficient and cost-effective than collecting real-world data in many cases. It can also be created on demand based on specifications rather than needing to wait to collect data once it occurs in reality. Synthetic data can also complement real-world data so that testing can occur for every imaginable variable even there isn't a good example in the real data set. This allows organizations to accelerate the testing of system performance and training of new systems.

The limitations for using real data for learning and testing are reduced when using fabricated data sets. Recent research suggests that it is possible to get the similar results as you would with authentic data sets.

4. AI/ML techniques used

There has been combination of multiple methods used for each Generator Validator and Evaluator as listed.

FOR NUMERIC DATA

Regression Strategies	Large Datasets (>50K)	Smaller Datasets (<50K)
Generator	**. Variational Autoencoder (VAE)**	**. Use inverse transform** **- PCA**
Validator	**. Artificial Neural Networks (ANN)** **. Autoencoder (AE)**	**. NB** **. Linear Regression** **. Random Forest** **. XGBoost**
Evaluator	**. Artificial Neural Networks (ANN)**	**. Linear Regression**

FOR IMAGES

Classification Strategies	Large Datasets (>50K)	Smaller Datasets (<50K)
Generator	. Variational Autoencoder (VAE)	Use inverse transform - PCA - NMF
Validator	. Convolutional Neural Networks (CNN) . Autoencoder (AE)	. XGBoost . Random Forest . ANN
Evaluator	. Convolutional Neural Networks (CNN)	. Convolutional Neural Networks (CNN)

5. PROJECT SOLUTION

The project was done in 2 segments considering the requirement for image and numeric data.

Module 1: Numeric Synthetic data
Module 2: Image Synthetic data

The following upcoming modules will dig in details of the project.

5.1. URL to access the webpage:

http://101.127.128.81:8080

6. Module 1 : Numeric Synthetic data

Life expectancy is one of the most important factors in end-of-life decision making. Good prognostication for example helps to determine the course of treatment and helps to anticipate the procurement of health care services and facilities, or more broadly: facilitates Advance Care Planning. Advance Care Planning improves the quality of the final phase of life by

stimulating doctors to explore the preferences for end-of-life care with their patients, and people close to the patients. Physicians, however, tend to overestimate life expectancy, and miss the window of opportunity to initiate Advance Care Planning. This research tests the creation of synthetic data for hospitals.

Life Expectancy: Exploratory Data Analysis
Goal: Find a set of features that affect Life Expectancy.
1. Data Cleaning
2. Data Exploration
3. Feature Engineering
4. Summary

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio	Total expenditure	Diphtheria	HIV/AIDS	GDP	Population	thinness 1-19 years	thinness 5-9 years	Income composition of resources	Schooling
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	19.1	83	6.0	8.16	65.0	0.1	584.259210	33736494.0	17.2	17.3	0.479	10.1
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	18.6	86	58.0	8.18	62.0	0.1	612.696514	327582.0	17.5	17.5	0.476	10.0
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	18.1	89	62.0	8.13	64.0	0.1	631.744976	31731688.0	17.7	17.7	0.470	9.9
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	17.6	93	67.0	8.52	67.0	0.1	669.959000	3696958.0	17.9	18.0	0.463	9.8
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	17.2	97	68.0	7.87	68.0	0.1	63.537231	2978599.0	18.2	18.2	0.454	9.5

6.1. Data Cleaning

In order to properly clean the data, it is important to understand the variables presented in the data. There are a number of things important to know about each variable: 1. What does the variable mean and what type of variable is it (Nominal/Ordinal/Interval/Ratio)? 2. Does the variable have missing values? If so, what should be done about them? 3. Does the variable have outliers? If so, what should be done about them?
Each of these questions will be answered in turn for all the variables. And those answers can be found in this section.

6.2. Dataset Description/Variable Descriptions

Dataset Description
This dataset is comprised of data from all over the world from various countries aggregated by the World Health Organization (WHO for short). The data is an aggregate of many indicators for a particular country in a particular year. In essence, the data is multiple indicators in a time series separated by country. A more in depth look into the context, content, acknowledgments, and inspiration for this dataset can be found [here](https://www.kaggle.com/kumarajarshi/life-expectancy-who).

Before getting into the variable descriptions, the string values for the columns/variables themselves are not very 'clean' so the following is a quick cleaning of the column/variable titles.

6.3. Variable Descriptions

Format: variable (type) – description

- country (Nominal) - the country in which the indicators are from (i.e. United States of America or Congo)

- year (Ordinal) - the calendar year the indicators are from (ranging from 2000 to 2015)
- status (Nominal) - whether a country is considered to be 'Developing' or 'Developed' by WHO standards
- life_expectancy (Ratio) - the life expectancy of people in years for a particular country and year
- adult_mortality (Ratio) - the adult mortality rate per 1000 population (i.e. number of people dying between 15 and 60 years per 1000 population); if the rate is 263 then that means 263 people will die out of 1000 between the ages of 15 and 60; another way to think of this is that the chance an individual will die between 15 and 60 is 26.3%
- infant_deaths (Ratio) - number of infant deaths per 1000 population; similar to above, but for infants
- alcohol (Ratio) - a country's alcohol consumption rate measured as liters of pure alcohol consumption per capita
- percentage_expenditure (Ratio) - expenditure on health as a percentage of Gross Domestic Product (gdp)
- hepatitis_b (Ratio) - number of 1 year olds with Hepatitis B immunization over all 1 year olds in population
- measles (Ratio) - number of reported Measles cases per 1000 population
- bmi (Interval/Ordinal) - average Body Mass Index (BMI) of a country's total population
- under-five_deaths (Ratio) - number of people under the age of five deaths per 1000 population
- polio (Ratio) - number of 1 year olds with Polio immunization over the number of all 1 year olds in population
- total_expenditure (Ratio) - government expenditure on health as a percentage of total government expenditure
- diphtheria (Ratio) - Diphtheria tetanus toxoid and pertussis (DTP3) immunization rate of 1 year olds
- hiv/aids (Ratio) - deaths per 1000 live births caused by HIV/AIDS for people under 5; number of people under 5 who die due to HIV/AIDS per 1000 births
- gdp (Ratio) - Gross Domestic Product per capita
    - population (Ratio) - population of a country
    - thinness_1-19_years (Ratio) - rate of thinness among people aged *10-19* (Note: variable should be renamed to *thinness_10-19_years* to more accurately represent the variable)
    - thinness_5-9_years (Ratio) - rate of thinness among people aged 5-9
    - income_composition_of_resources (Ratio) - Human Development Index in terms of income composition of resources (index ranging from 0 to 1)
    - schooling (Ratio) - average number of years of schooling of a population

As stated above it would be useful to change the name of the variable `thinness_1-19_years` to `thinness_10-19_years` as it is a more accurate depiction of what the variable means.

Now that the descriptions of the dataset and variables have been made, a look at the missing values of each variable should be done.

6.4. Missing Values

There are few things that must be done concerning missing values:
1. Detection of missing values
   - Find nulls
   - Could a null be signified by anything other than null? Zero values perhaps?
2. Dealing with missing values
   - Fill nulls? Impute or Interpolate
   - Eliminate nulls?

6.5. Missing Values Detection

Finding possible inexplicit nulls
These nulls would be missing values that aren't necessarily easy to find using the `df.info()` method.
- What values could be null?
- What values could be erroneous?

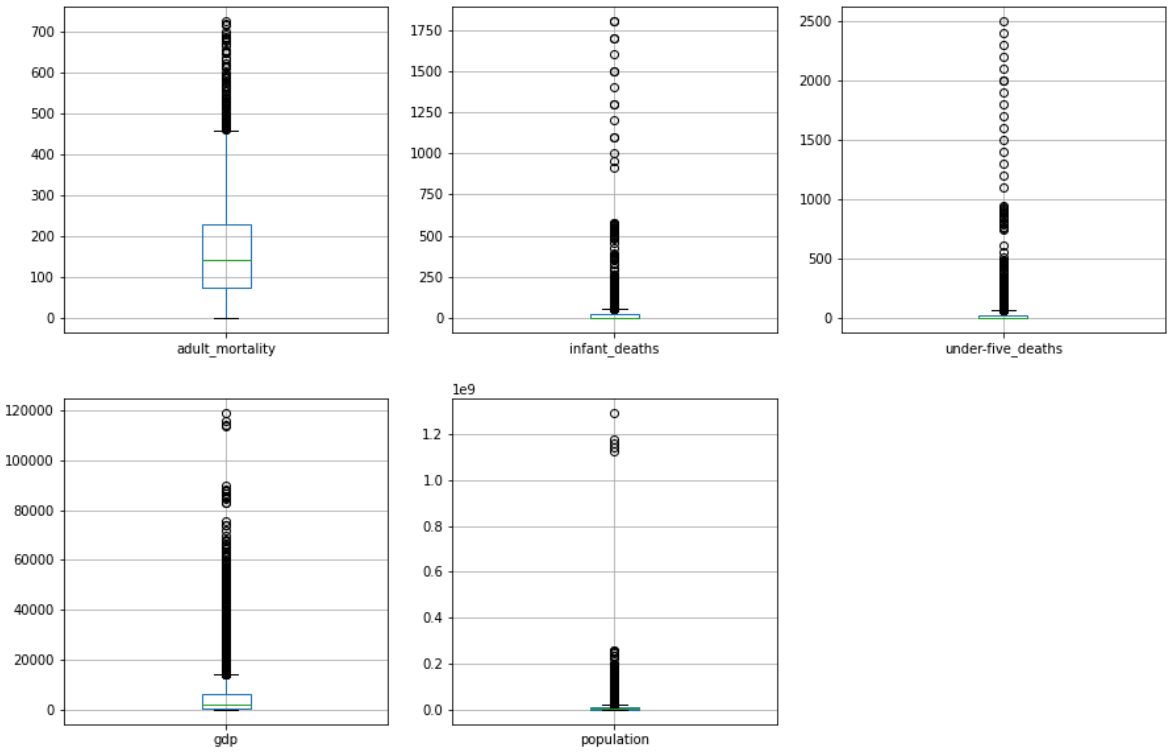
Inexplicit Nulls
The easiest and quickest method here would be to do a quick `df.describe()` and look at each variable on its own to see if the values make sense given the description of the variable.

	count	mean	std	min	25%	50%	75%	max
life_expectancy	2928.0	69.0	10.0	36.0	63.0	72.0	76.0	8.900000e+01
adult_mortality	2928.0	165.0	124.0	1.0	74.0	144.0	228.0	7.230000e+02
infant_deaths	2938.0	30.0	118.0	0.0	0.0	3.0	22.0	1.800000e+03
alcohol	2744.0	5.0	4.0	0.0	1.0	4.0	8.0	1.800000e+01
percentage_expenditure	2938.0	738.0	1988.0	0.0	5.0	65.0	442.0	1.948000e+04
hepatitis_b	2385.0	81.0	25.0	1.0	77.0	92.0	97.0	9.900000e+01
measles	2938.0	2420.0	11467.0	0.0	0.0	17.0	360.0	2.121830e+05
bmi	2904.0	38.0	20.0	1.0	19.0	44.0	56.0	8.700000e+01
under-five_deaths	2938.0	42.0	160.0	0.0	0.0	4.0	28.0	2.500000e+03
polio	2919.0	83.0	23.0	3.0	78.0	93.0	97.0	9.900000e+01
total_expenditure	2712.0	6.0	2.0	0.0	4.0	6.0	7.0	1.800000e+01
diphtheria	2919.0	82.0	24.0	2.0	78.0	93.0	97.0	9.900000e+01
hiv/aids	2938.0	2.0	5.0	0.0	0.0	0.0	1.0	5.100000e+01
gdp	2490.0	7483.0	14270.0	2.0	464.0	1767.0	5911.0	1.191730e+05
population	2286.0	12753375.0	61012097.0	34.0	195793.0	1386542.0	7420359.0	1.293859e+09
thinness_10-19_years	2904.0	5.0	4.0	0.0	2.0	3.0	7.0	2.800000e+01
thinness_5-9_years	2904.0	5.0	5.0	0.0	2.0	3.0	7.0	2.900000e+01
income_composition_of_resources	2771.0	1.0	0.0	0.0	0.0	1.0	1.0	1.000000e+00
schooling	2775.0	12.0	3.0	0.0	10.0	12.0	14.0	2.100000e+01

Things that may not make sense from above:

- Adult mortality of 1? This is likely an error in measurement, but what values make sense here? May need to change to null if under a certain threshold.

- Infant deaths as low as 0 per 1000? That just isn't plausible - I'm deeming those values to actually be null. Also on the other end 1800 is likely an outlier, but it is possible in a country with very high birthrates and perhaps a not very high population total - this can be dealt with later.
- BMI of 1 and 87.3? Pretty sure the whole population would not exist if that were the case. A BMI of 15 or lower is seriously underweight and a BMI of 40 or higher is morbidly obese, therefore a large number of these measurements just seem unrealistic...this variable might not be worth digging into at all.
- Under Five Deaths, similar to infant deaths just isn't likely (perhaps even impossible) to have values at zero.
- GDP per capita as low as 1.68 (USD) possible? Doubtful - but perhaps values this low are outliers.
- Population of 34 for an entire country?



There are a few of the above that could simply be outliers, but there are some that almost certainly have to be errors of some sort. Of the above variables, changes to null will be made for the following since these numbers don't make any sense:

1. Adult mortality rates lower than the 5th percentile
2. Infant deaths of 0
3. BMI less than 10 and greater than 50
4. Under Five deaths of 0

```
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   country                       2938 non-null   object
 1   year                          2938 non-null   int64
 2   status                        2938 non-null   object
 3   life_expectancy               2928 non-null   float64
 4   adult_mortality               2928 non-null   float64
 5   infant_deaths                 2938 non-null   int64
 6   alcohol                       2744 non-null   float64
 7   percentage_expenditure        2938 non-null   float64
 8   hepatitis_b                   2385 non-null   float64
 9   measles                       2938 non-null   int64
 10  bmi                           2904 non-null   float64
 11  under-five_deaths             2938 non-null   int64
 12  polio                         2919 non-null   float64
 13  total_expenditure             2712 non-null   float64
 14  diphtheria                    2919 non-null   float64
 15  hiv/aids                      2938 non-null   float64
 16  gdp                           2490 non-null   float64
 17  population                    2286 non-null   float64
 18  thinness_10-19_years          2904 non-null   float64
 19  thinness_5-9_years            2904 non-null   float64
```

6.6. Null Breakdown Analysis

```
[row 3] life_expectancy has 10 null values: 0.34% null
[row 4] adult_mortality has 155 null values: 5.28% null
[row 5] infant_deaths has 848 null values: 28.86% null
[row 6] alcohol has 194 null values: 6.6% null
[row 8] hepatitis_b has 553 null values: 18.82% null
[row 10] bmi has 34 null values: 1.16% null
[row 11] under-five_deaths has 785 null values: 26.72% null
[row 12] polio has 19 null values: 0.65% null
[row 13] total_expenditure has 226 null values: 7.69% null
[row 14] diphtheria has 19 null values: 0.65% null
[row 16] gdp has 448 null values: 15.25% null
[row 17] population has 652 null values: 22.19% null
[row 18] thinness_10-19_years has 34 null values: 1.16% null
[row 19] thinness_5-9_years has 34 null values: 1.16% null
[row 20] income_composition_of_resources has 167 null values: 5.68% null
[row 21] schooling has 163 null values: 5.55% null
Out of 22 total columns, 16 contain null values; 72.73% columns contain null values.
```

6.7. Dealing with Missing Values

Nearly half of the BMI variable's values are null, it is likely best to remove this variable altogether.

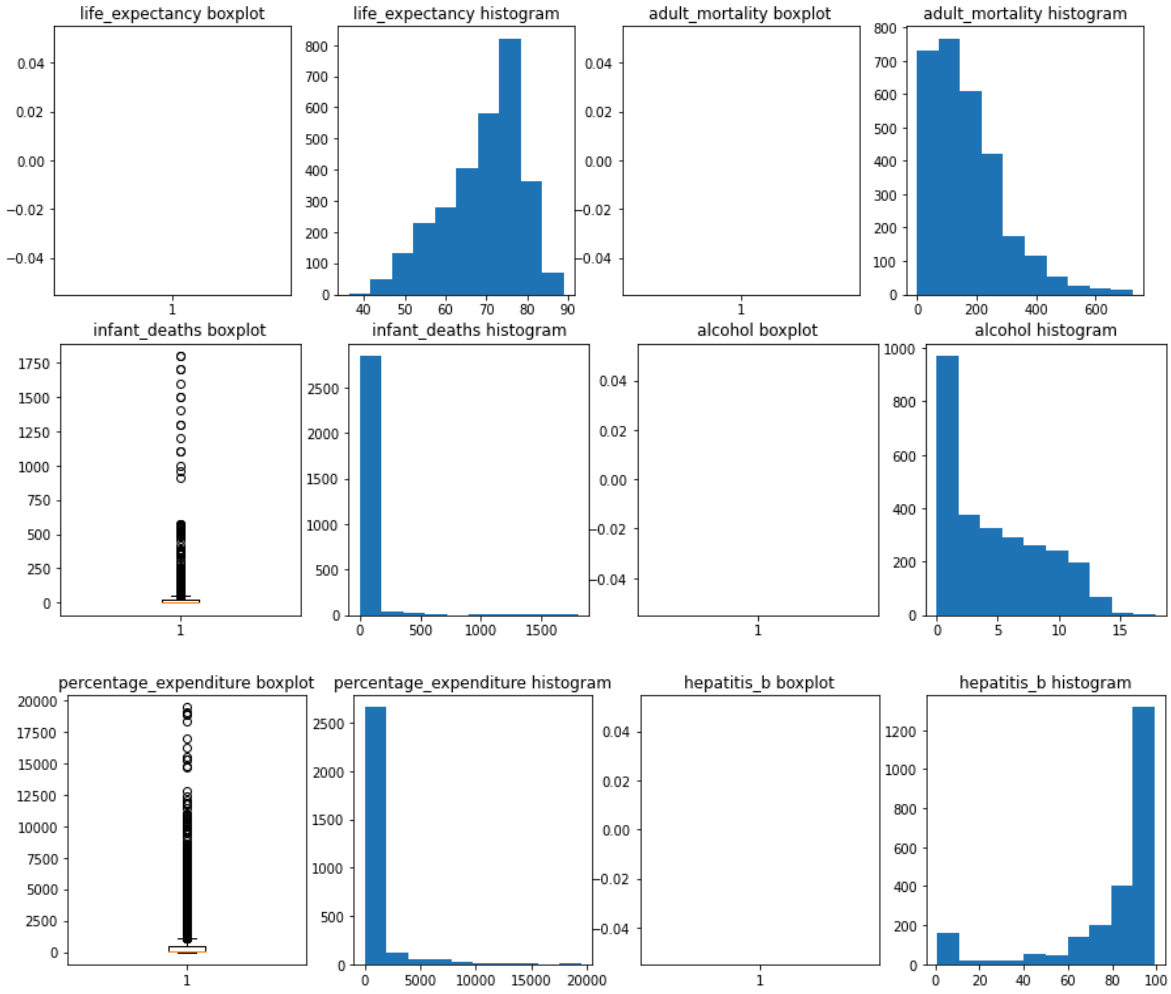
Alright, so it looks like there are a lot of columns containing null values, since this is time series data assorted by country, the best course of action would be to interpolate the data by country. However, when attempting to interpolate by country it doesn't fill in any values as the countries' data for all the null values are null for each year, therefore imputation by year may be the best possible method here. Imputation of each year's mean is done below.

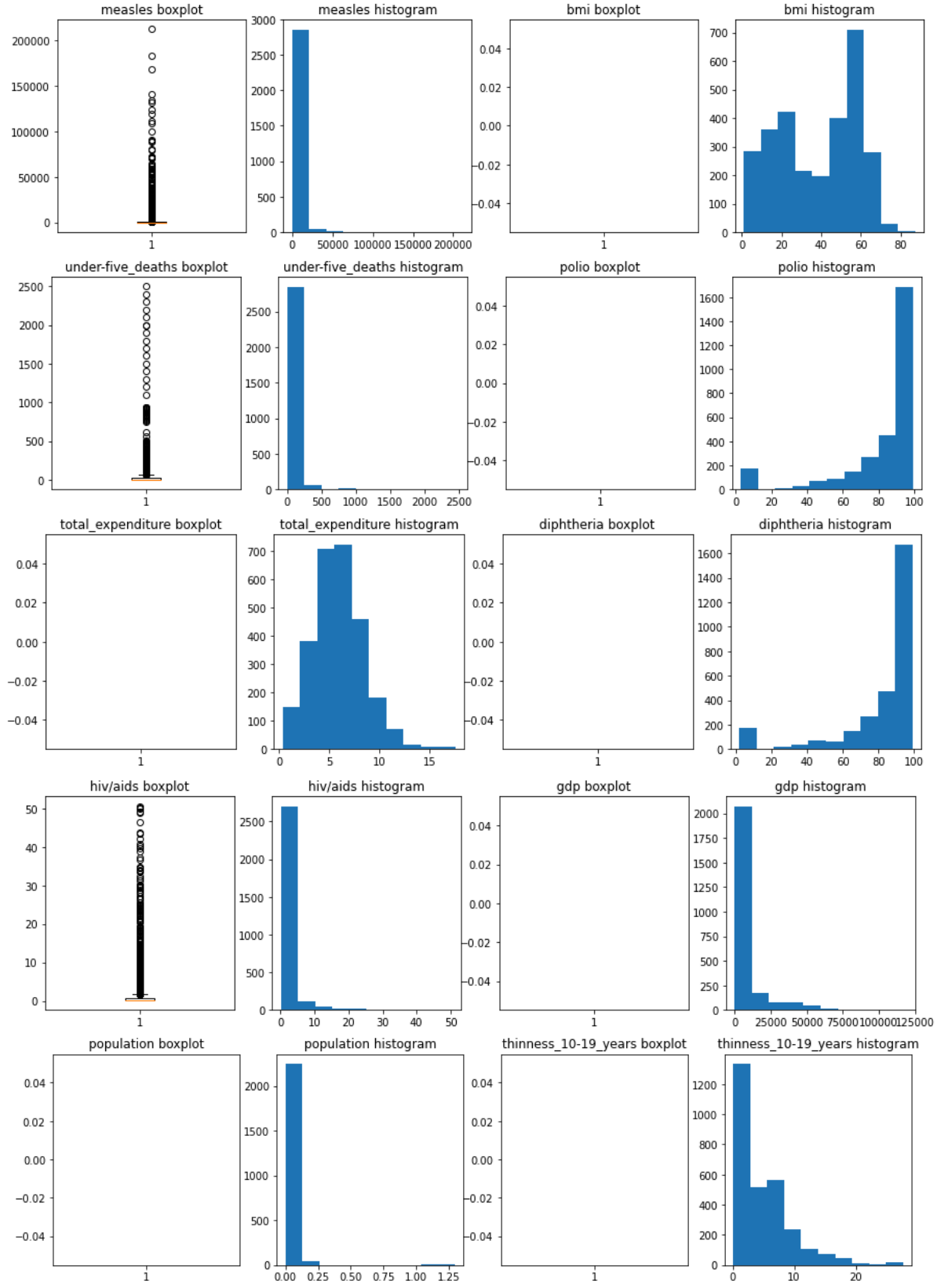
6.8. Outliers

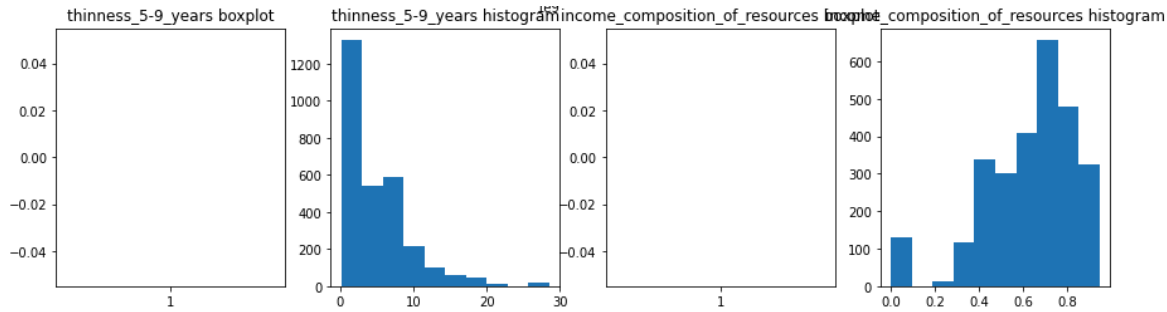
Similar to missing values, there are a few things that need done in order to deal with outliers:
1. Detect the outliers
   - Boxplots/histograms
   - Tukey's Method
2. Deal with outliers
   - Drop outliers?
   - Limit/Winsorize outliers?
   - Transform the data using log/inverse/square root/etc?

6.8.1. Outliers Detection

First a boxplot and histogram will be created for each continuous variable in order to visually see if outliers exist.




measles boxplot
measles histogram
bmi boxplot
bmi histogram
under-five_deaths boxplot
under-five_deaths histogram
polio boxplot
polio histogram
total_expenditure boxplot
total_expenditure histogram
diphtheria boxplot
diphtheria histogram
hiv/aids boxplot
hiv/aids histogram
gdp boxplot
gdp histogram
population boxplot
population histogram
thinness_10-19_years boxplot
thinness_10-19_years histogram



Visually, it is plain to see that there are a number of outliers for all of these variables – including the target variable, life expectancy. The same will be done statistically using Tukey's method below – outliers being considered anything outside of 1.5 times the IQR.

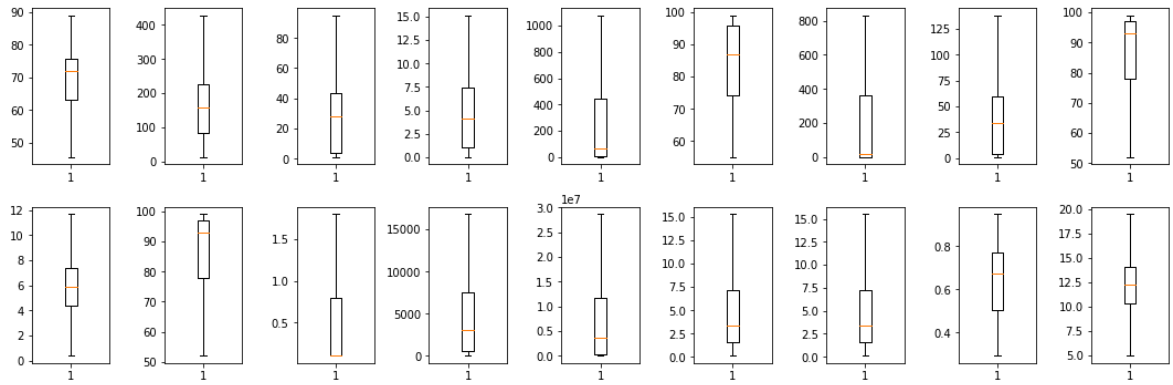
```
---------------life_expectancy---------------
Number of outliers: 17
Percent of data that is outlier: 0.58%
---------------adult_mortality---------------
Number of outliers: 97
Percent of data that is outlier: 3.3%
---------------infant_deaths---------------
Number of outliers: 135
Percent of data that is outlier: 4.59%
---------------alcohol---------------
Number of outliers: 3
Percent of data that is outlier: 0.1%
---------------percentage_expenditure---------------
Number of outliers: 389
Percent of data that is outlier: 13.24%
---------------hepatitis_b---------------
Number of outliers: 222
Percent of data that is outlier: 7.56%
---------------measles---------------
Number of outliers: 542
Percent of data that is outlier: 18.45%
---------------under-five_deaths---------------
Number of outliers: 142
Percent of data that is outlier: 4.83%
---------------polio---------------
```

6.8.3. Dealing with Outliers

There are a number of ways to deal with outliers in a dataset, the usual options are as follows:
1. Drop Outliers (best avoided in order to keep as much information as possible)
2. Limit values to upper and/or lower bounds (Winsorize the data)
3. Transform the data (log/inverse/square root/etc.)
   - advantage: can 'normalize' the data and eliminate outliers
   - disadvantage: cannot be done to variables containing values of 0 or below

Since each variable has a unique amount of outliers and also has outliers on different sides of the data, the best route to take is probably winsorizing (limiting) the values for each variable on its own until no outliers remain. The function below allows me to do exactly that by going variable by variable with the ability to use a lower limit and/or upper limit for winsorization. By default the function will show two boxplots side by side for the variable (one boxplot of the original data, and one with the winsorized change). Once a satisfactory limit is found (by visual analysis), the winsorized data will be saved in the `wins_dict` dictionary so the data can easily be accessed later.

All the variables have now been winsorized as little as possible in order to keep as much data in tact as possible while still being able to eliminate the outliers. Finally, small boxplots will be shown for each variable's winsorized data to show that the outliers have indeed been dealt with.



6.9. Data Exploration

With that out of the way, the main areas of interest in this section are as follows:
1. Univariate Analysis
   - Continuous variables
   - Categorical Variables
2. Bivariate Analysis
   - Continuous to Continuous variables
   - Continuous to Categorical variables
   - Categorical to Categorical variables

6.10. Univariate Analysis

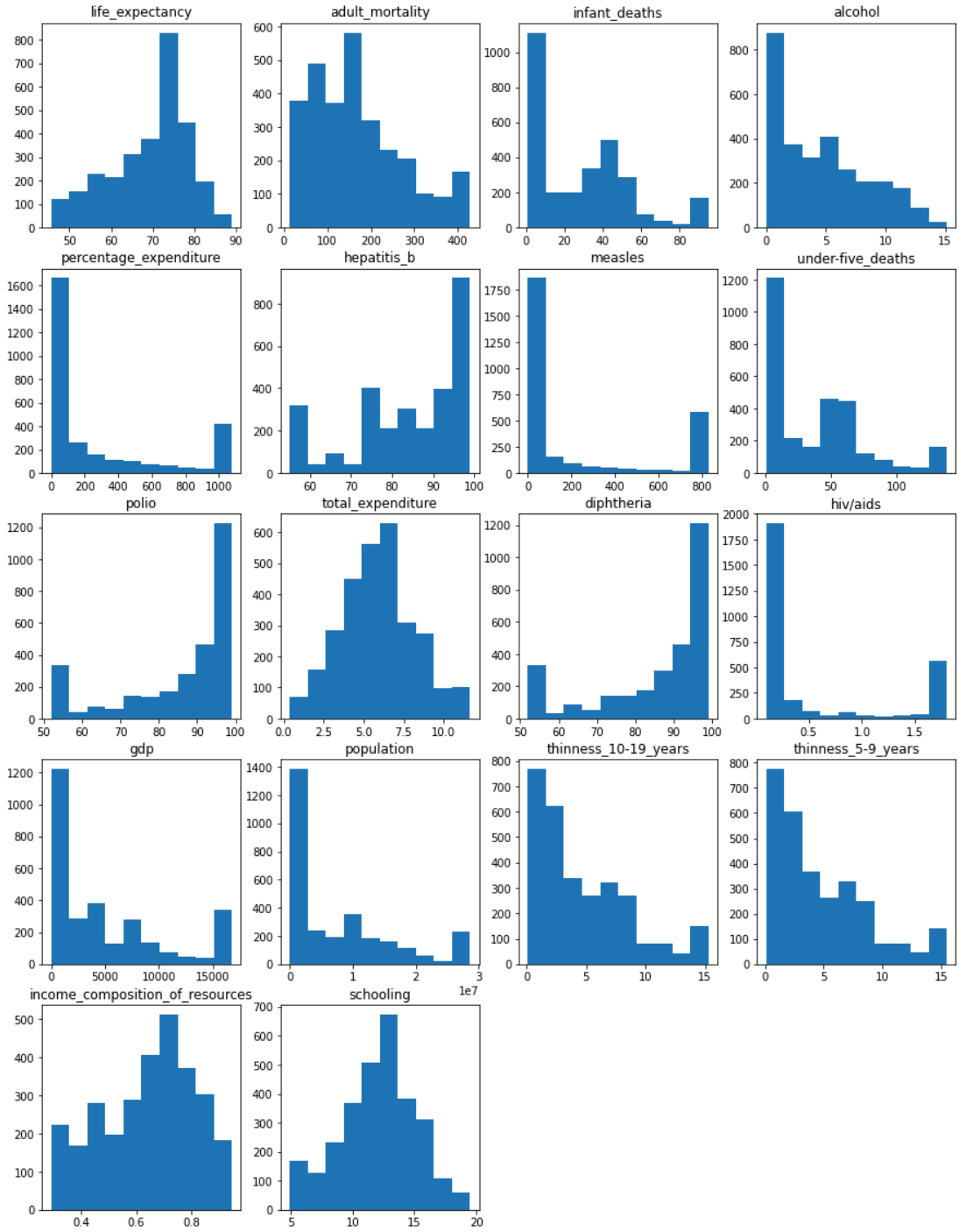
Univariate analysis is looking at the data for each variable on its own. This is generally done best by using histograms for continuous data, count/barplots for categorical data and of course by getting the descriptive stats by using

Descriptive Statistics

```
wins_df.describe().transpose().round(0)
```

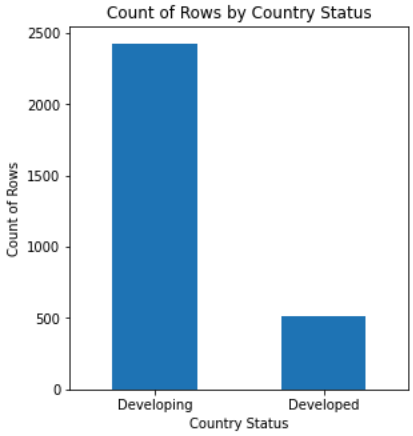
	count	mean	std	min	25%	50%	75%	max
year	2938.0	2008.0	5.0	2000.0	2004.0	2008.0	2012.0	2015.0
life_expectancy	2938.0	69.0	9.0	46.0	63.0	72.0	76.0	89.0
adult_mortality	2938.0	169.0	108.0	13.0	84.0	157.0	227.0	428.0
infant_deaths	2938.0	29.0	26.0	1.0	4.0	28.0	44.0	95.0
alcohol	2938.0	5.0	4.0	0.0	1.0	4.0	7.0	15.0
percentage_expenditure	2938.0	282.0	384.0	0.0	5.0	65.0	442.0	1078.0
hepatitis_b	2938.0	84.0	14.0	55.0	74.0	87.0	96.0	99.0
measles	2938.0	221.0	329.0	0.0	0.0	17.0	360.0	831.0
under-five_deaths	2938.0	39.0	38.0	1.0	4.0	34.0	60.0	138.0
polio	2938.0	86.0	15.0	52.0	78.0	93.0	97.0	99.0
total_expenditure	2938.0	6.0	2.0	0.0	4.0	6.0	7.0	12.0
diphtheria	2938.0	85.0	15.0	52.0	78.0	93.0	97.0	99.0
hiv/aids	2938.0	1.0	1.0	0.0	0.0	0.0	1.0	2.0
gdp	2938.0	5034.0	5409.0	2.0	580.0	3117.0	7464.0	16784.0
population	2938.0	7508368.0	8646842.0	34.0	418917.0	3675929.0	11813315.0	28656282.0
thinness_10-19_years	2938.0	5.0	4.0	0.0	2.0	3.0	7.0	15.0
thinness_5-9_years	2938.0	5.0	4.0	0.0	2.0	3.0	7.0	16.0
income_composition_of_resources	2938.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0
schooling	2938.0	12.0	3.0	5.0	10.0	12.0	14.0	20.0

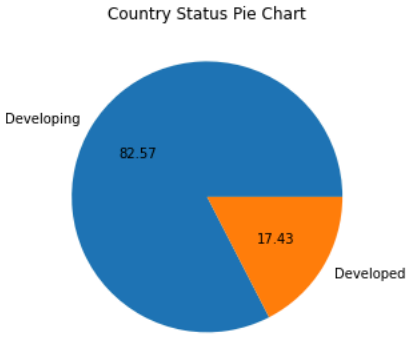
Visual Distributions



The winsorization had a large effect on some variables while not having too much of an effect on others. Even though all of these variables were winsorized in some fashion, some variables are much more obviously winsorized than others. What about the categorical variables, how many of each of these are there in the data (in essence, what is their distribution?)

Again, not the most useful plot, but does display that each year has the same amount of rows, except for 2013, which contains 10 more rows than the rest (the countries with only one row from the prior graph's data must be from 2013 alone). This shouldn't have a detrimental effect on analysis.





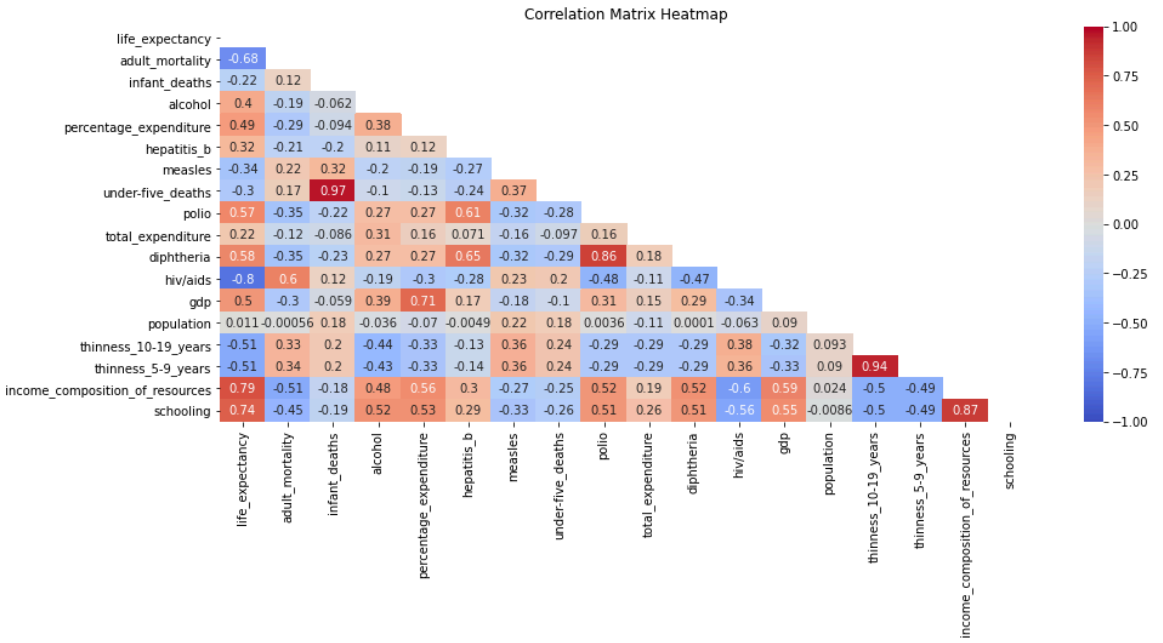
This graph, though simple, is important. The above displays that the majority of our data comes from countries listed as 'Developing' - 82.57% to be exact. It is likely that any model used will more accurately depict results for 'Developing' countries over 'Developed' countries as the majority of the data lies within countries that are 'Developing' rather than 'Developed'.

6.11. Bivariate Analysis

There are a number of things that should be examined here:
1. Continuous variables compared to the life expectancy (target variable) and to one another
2. Categorical variables compared to the life expectancy (target variable)
3. Comparison of Country Status and Year to Continuous variables (country has an extremely large number of values with small sample sizes, so country comparisons aren't especially helpful for this dataset)

6.11.1. Continuous to Continuous Analysis

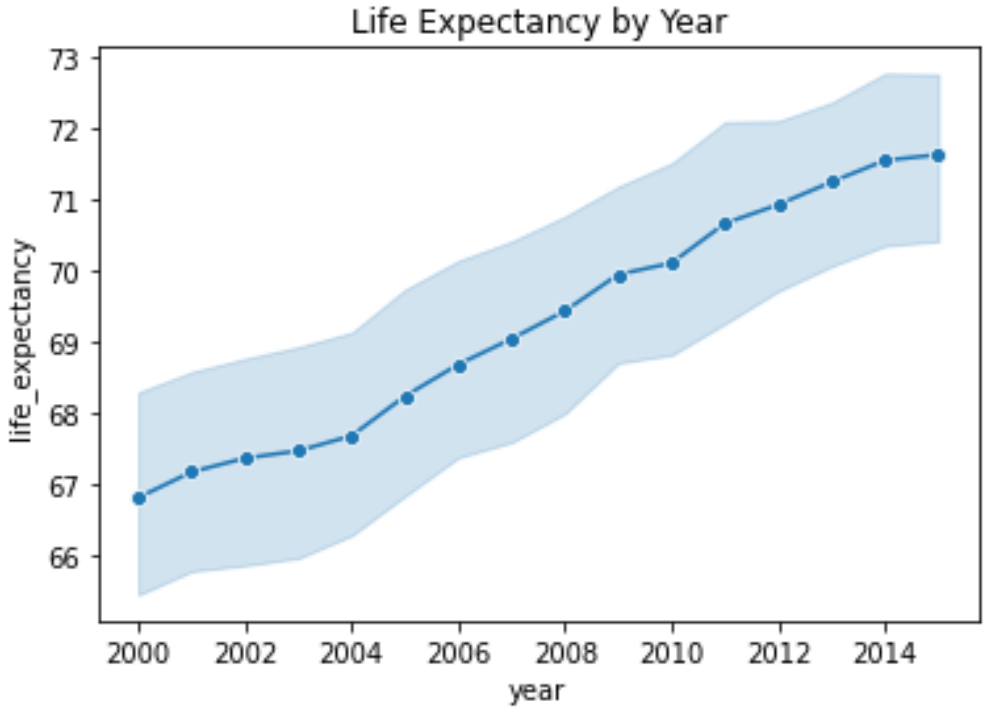
	life_expectancy	adult_mortality	infant_deaths	alcohol	percentage_expenditure	hepatitis_b	measles	under-five_deaths	polio	total_expenditure	diphtheria	hiv/aids	gdp
life_expectancy	1.000000	-0.677680	-0.222292	0.395089	0.488440	0.315120	-0.337887	-0.298626	0.573291	0.222306	0.578952	-0.796939	0.501011
adult_mortality	-0.677680	1.000000	0.119906	-0.187971	-0.293870	-0.214673	0.216200	0.168908	-0.348198	-0.121052	-0.351136	0.596271	-0.297774
infant_deaths	-0.222292	0.119906	1.000000	-0.062119	-0.094053	-0.196145	0.323610	0.966996	-0.217511	-0.086289	-0.225997	0.124310	-0.059337
alcohol	0.395089	-0.187971	-0.062119	1.000000	0.378069	0.106352	-0.197193	-0.104954	0.265830	0.308434	0.272948	-0.193443	0.389598
percentage_expenditure	0.488440	-0.293870	-0.094053	0.378069	1.000000	0.122870	-0.194687	-0.131361	0.268385	0.159830	0.268411	-0.295791	0.712940
hepatitis_b	0.315120	-0.214673	-0.196145	0.106352	0.122870	1.000000	-0.266487	-0.239049	0.608008	0.071100	0.647198	-0.278977	0.172066
measles	-0.337887	0.216200	0.323610	-0.197193	-0.194687	-0.266487	1.000000	0.368517	-0.320104	-0.161737	-0.315164	0.226305	-0.183792
under-five_deaths	-0.298626	0.168908	0.966996	-0.104954	-0.131361	-0.239049	0.368517	1.000000	-0.281667	-0.096826	-0.291057	0.198160	-0.102372
polio	0.573291	-0.348198	-0.217511	0.265830	0.268385	0.608008	-0.320104	-0.281667	1.000000	0.164149	0.855849	-0.475611	0.309033
total_expenditure	0.222306	-0.121052	-0.086289	0.308434	0.159830	0.071100	-0.161737	-0.096826	0.164149	1.000000	0.176715	-0.110629	0.146954
diphtheria	0.578952	-0.351136	-0.225997	0.272948	0.268411	0.647198	-0.315164	-0.291057	0.855849	0.176715	1.000000	-0.474643	0.290940
hiv/aids	-0.796939	0.596271	0.124310	-0.193443	-0.295791	-0.278977	0.226305	0.198160	-0.475611	-0.110629	-0.474643	1.000000	-0.335518
gdp	0.501011	-0.297774	-0.059337	0.389598	0.712940	0.172066	-0.183792	-0.102372	0.309033	0.146954	0.290940	-0.335518	1.000000
population	0.011363	-0.000562	0.176352	-0.036283	-0.070366	-0.004936	0.222488	0.180288	0.003618	-0.105257	0.000104	-0.062750	0.090051
thinness_10-19_years	-0.514966	0.332904	0.196669	-0.436755	-0.331693	-0.134103	0.359571	0.235471	-0.288660	-0.285700	-0.294792	0.375062	-0.324997
thinness_5-9_years	-0.512694	0.339266	0.202573	-0.427166	-0.333748	-0.139528	0.364201	0.236927	-0.287206	-0.294560	-0.288930	0.357066	-0.332025
composition_of_resources	0.792878	-0.511340	-0.178389	0.475112	0.556006	0.299493	-0.274976	-0.253539	0.515361	0.185535	0.516429	-0.598122	0.588686
schooling	0.742306	-0.450785	-0.191062	0.519757	0.530422	0.290399	-0.329274	-0.264996	0.509788	0.259746	0.511045	-0.557965	0.547028



The above heatmap is very useful! It very easily displays a number of important correlations between variables. Some general takeaways from the graphic above:

- Life Expectancy (target variable) appears to be relatively highly correlated (negatively or positively) with:
  - Adult Mortality (negative)
  - HIV/AIDS (negative)
  - Income Composition of Resources (positive)
  - Schooling (positive)
- Life expectancy (target variable) is extremely lowly correlated to population (nearly no correlation at all)
- Infant deaths and Under Five deaths are extremely highly correlated
- Percentage Expenditure and GDP are relatively highly correlated
- Hepatitis B vaccine rate is relatively positively correlated with Polio and Diphtheria vaccine rates
- Polio vaccine rate and Diphtheria vaccine rate are very positively correlated
- HIV/AIDS is relatively negatively correlated with Income Composition of Resources
- Thinness of 5-9 Year olds rate and Thinness of 10-15 Year olds rate is extremely highly correlated
- Income Composition of Resources and Schooling are very highly correlated

6.11.2. Categorical to Life Expectancy Comparison



6.12. Feature Engineering

First off, since it is apparent that the status of a country should be included in some way in the final features of the data, one hot encoding will be conducted in order to include it in the future model.

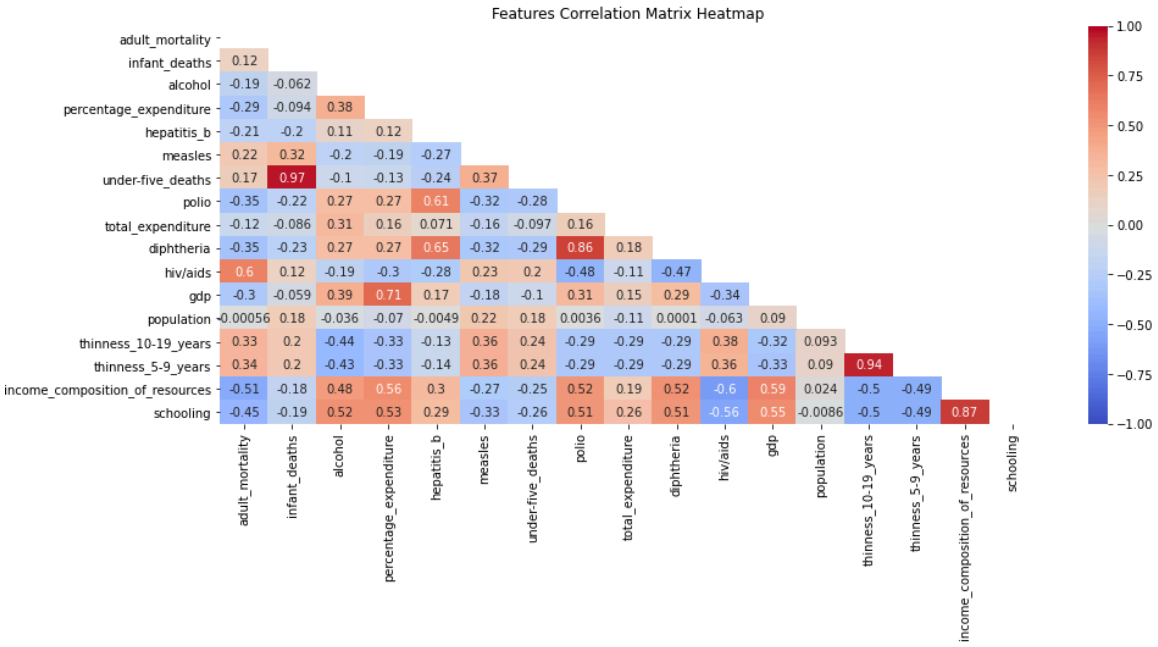
	Developed	Developing
life_expectancy	0.483121	-0.483121
adult_mortality	-0.310207	0.310207
infant_deaths	-0.011644	0.011644
alcohol	0.580249	-0.580249
percentage_expenditure	0.420621	-0.420621
hepatitis_b	0.108121	-0.108121
measles	-0.130744	0.130744
under-five_deaths	-0.045339	0.045339
polio	0.265064	-0.265064
total_expenditure	0.284829	-0.284829
diphtheria	0.267119	-0.267119
hiv/aids	-0.290242	0.290242
gdp	0.434245	-0.434245
population	-0.058526	0.058526
thinness_10-19_years	-0.395916	0.395916
thinness_5-9_years	-0.396833	0.396833
income_composition_of_resources	0.510650	-0.510650
schooling	0.508211	-0.508211
Developed	1.000000	-1.000000
Developing	-1.000000	1.000000

From the above it can be observed that whether a country is 'Developed' or not is certainly correlated with a number of variables, but not extremely highly. However, it does have a very low correlation with Infant Deaths, Under Five Deaths and Population.

Next, the categorical columns, 'year' and 'country' will be dropped as they don't have significant differences among life expectancy.

	adult_mortality	infant_deaths	alcohol	percentage_expenditure	hepatitis_b	measles	under-five_deaths	polio	total_expenditure	diphtheria	hiv/aids	gdp	population	thinness_10-19_years	thinness_5-9_years
0	263.0	62.000000	0.010000	71.279624	65.0	831	83.000000	52.0	8.16	65.0	0.1	584.259210	2.865628e+07	15.3	15.5
16	74.0	35.129032	4.600000	364.975229	99.0	0	44.844961	99.0	6.00	99.0	0.1	3954.227830	2.887300e+04	1.2	1.3
32	19.0	21.000000	5.288333	0.000000	95.0	63	24.000000	95.0	7.08	95.0	0.1	4132.762920	2.865628e+07	6.0	5.8
48	335.0	66.000000	5.288333	0.000000	64.0	118	98.000000	52.0	7.08	64.0	1.8	3695.793748	2.785935e+06	8.3	8.2
64	13.0	35.129032	5.288333	0.000000	99.0	0	44.844961	86.0	7.08	99.0	0.2	13566.954100	1.109741e+07	3.3	3.3
80	116.0	8.000000	5.288333	0.000000	94.0	0	9.000000	93.0	7.08	94.0	0.1	13467.123600	2.865628e+07	1.0	0.9
96	118.0	1.000000	5.288333	0.000000	94.0	33	1.000000	96.0	7.08	94.0	0.1	369.654776	2.916950e+05	2.1	2.2
12	59.0	1.000000	5.288333	0.000000	93.0	74	1.000000	93.0	7.08	93.0	0.1	16784.346160	2.378934e+07	0.6	0.6
28	65.0	35.129032	5.288333	0.000000	93.0	309	44.844961	93.0	7.08	93.0	0.1	16784.346160	8.633169e+06	1.9	2.1
44	118.0	5.000000	5.288333	0.000000	96.0	0	6.000000	98.0	7.08	96.0	0.1	55.313820	9.649341e+06	2.8	2.9

From the prior analysis, there are a number of variables that are very or extremely highly correlated with one another. In those cases, the variable which is most highly correlated to Life Expectancy (target variable) will be kept while the others will be dismissed.



6.12.1. Principal Component Analysis

It may be useful to run a Principal Components Analysis (PCA) on this data to reduce the amount of dimensions (features). But there are a number of assumptions/requirements when it comes to PCA:

- Continuous data: the data used should be of a continuous type
- Sample size: the sample size should have between 5-10 samples per feature

- Normalized data: the data is generally normally distributed
- Correlation: there should be correlation between the features
- Linearity: it is assumed that relationships between features are linear
- Outliers: PCA is sensitive to outliers, therefore outliers should not be present

The features set currently satisfies 3 of the above assumptions: sample size, correlation, outliers. The linearity assumption may not be true, the data is not currently normalized and not all the data is continuous - the developed indicator is categorical. First the 'Developed' variable should be removed.

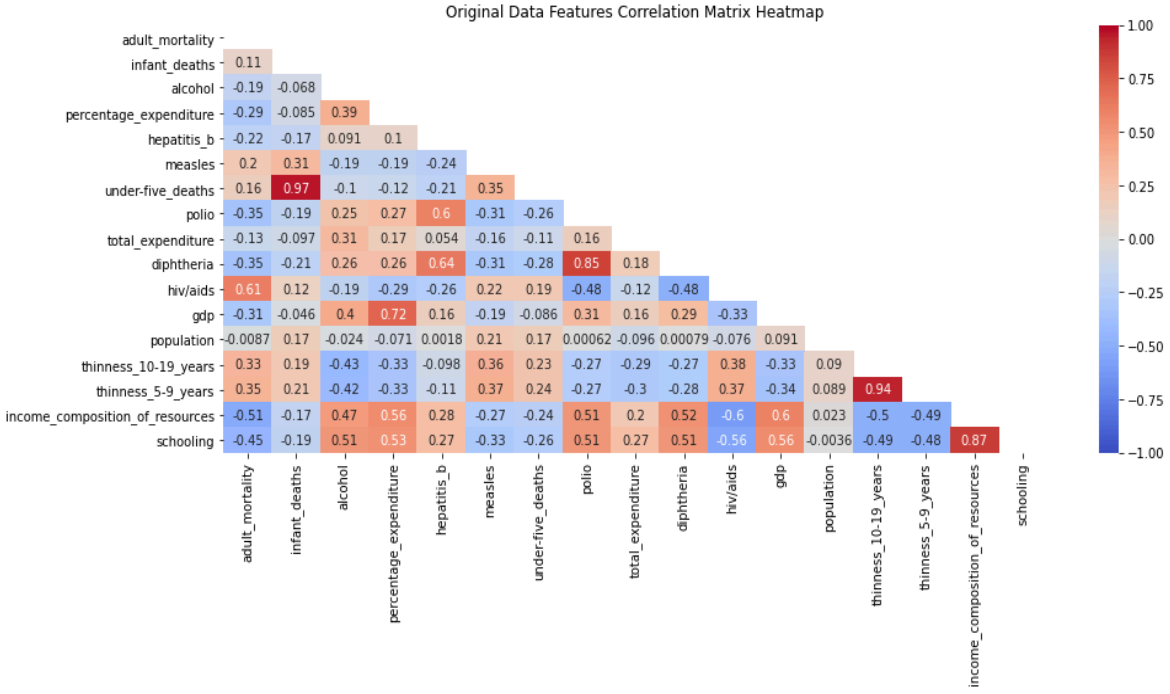
	pca1	pca2	pca3	pca4	pca5	pca6	pca7
0	-4.474150e+06	-4792.482181	591.008663	-24.127604	-8.154297	-60.513666	-13.224046
1	-5.693990e+06	-4234.147290	-217.692017	7.836378	-119.538307	-30.518219	-5.623202
2	-6.867430e+06	3515.555709	-186.310343	-569.550273	8.383688	-16.558842	-7.434941
3	-3.535642e+06	9450.328173	-3.856657	552.158141	29.808473	-36.435128	-5.693716
4	-2.095906e+06	-4943.478550	113.896692	26.067800	145.080133	-30.838492	23.809608
5	5.430133e+06	-4950.965954	-272.276223	53.776802	246.409613	-1.589566	18.586084
6	-6.407530e+06	-4691.113139	619.021203	-9.147291	104.651173	-10.285710	14.637855
7	-3.435464e+06	-4219.336794	596.356795	-29.617350	34.253935	45.979203	0.214170
8	-4.679089e+06	-3095.153697	-205.955324	32.626877	56.097383	-38.378828	-17.617551
9	-7.346096e+06	12154.136054	-25.788170	-134.576819	-20.938456	-24.285415	5.981242

	pca1	pca2	pca3	pca4	pca5	pca6	pca7
adult_mortality	-0.000000	-0.006221	0.059431	0.033605	0.995136	-0.047096	-0.050664
infant_deaths	0.000001	-0.000287	0.022602	0.000244	0.016570	0.561645	-0.096735
alcohol	-0.000000	0.000289	-0.001480	-0.001757	-0.001221	-0.001804	-0.016655
percentage_expenditure	-0.000003	0.052054	-0.118570	-0.990706	0.041160	0.004230	0.000918
hepatitis_b	0.000000	0.000397	-0.009581	0.002505	-0.020861	-0.048056	-0.499778
measles	0.000008	-0.012376	0.989768	-0.121717	-0.057842	-0.042999	-0.013212
under-five_deaths	0.000001	-0.000698	0.037926	0.000341	0.035949	0.818866	-0.057127
polio	0.000000	0.000864	-0.012795	-0.001577	-0.034394	-0.058156	-0.605678
total_expenditure	-0.000000	0.000070	-0.000903	-0.000411	-0.001383	-0.002068	-0.006952
diphtheria	0.000000	0.000820	-0.012999	-0.002003	-0.035801	-0.063674	-0.605370
hiv/aids	-0.000000	-0.000041	0.000415	0.000163	0.003387	0.001047	0.007283
gdp	0.000058	0.998547	0.018881	0.050350	0.003441	-0.000184	0.000642
population	1.000000	-0.000058	-0.000009	-0.000005	0.000000	-0.000001	0.000000
thinness_10-19_years	0.000000	-0.000250	0.003761	0.001030	0.007238	0.006945	0.002231
thinness_5-9_years	0.000000	-0.000256	0.003909	0.000961	0.007822	0.007471	0.001569
income_composition_of_resources	0.000000	0.000019	-0.000101	-0.000109	-0.000521	-0.000429	-0.001893
schooling	-0.000000	0.000319	-0.002425	-0.001823	-0.007543	-0.008417	-0.033895

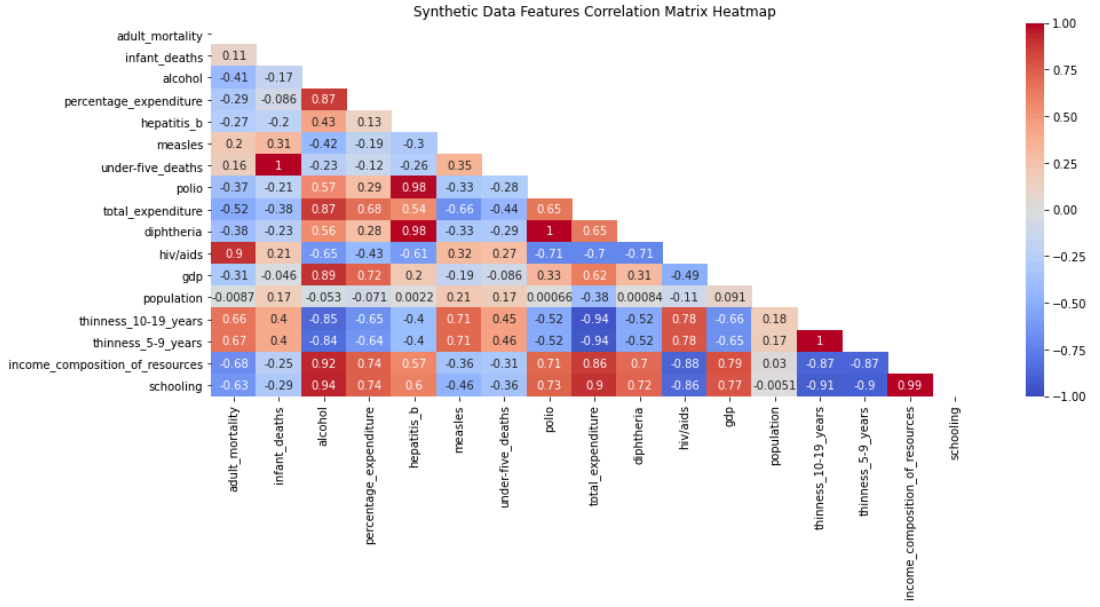
6.12.2. Summary Statistics of Original and Synthetic Dataset

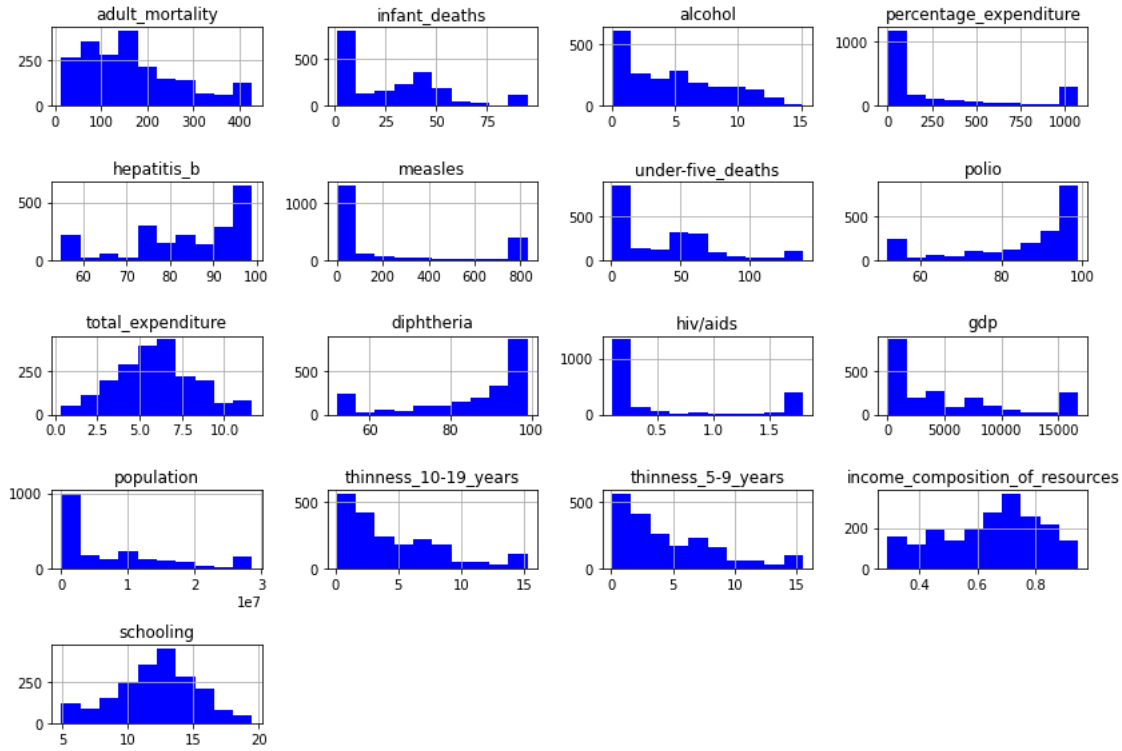
	count	mean	std	min	25%	50%	75%	max
adult_mortality	2056.0	167.84	108.36	13.00	82.00	153.00	225.00	428.00
infant_deaths	2056.0	28.17	25.46	1.00	4.00	27.00	43.77	95.00
alcohol	2056.0	4.70	3.89	0.01	1.17	4.22	7.55	15.14
percentage_expenditure	2056.0	286.02	387.43	0.00	5.26	66.55	459.50	1077.71
hepatitis_b	2056.0	83.82	13.69	55.00	74.10	87.00	96.00	99.00
measles	2056.0	216.51	326.58	0.00	0.00	16.50	336.75	831.00
under-five_deaths	2056.0	37.98	37.32	1.00	4.00	33.00	60.01	138.00
polio	2056.0	85.62	15.26	52.00	78.00	93.00	97.00	99.00
total_expenditure	2056.0	6.00	2.33	0.37	4.40	5.93	7.39	11.66
diphtheria	2056.0	85.64	15.13	52.00	78.00	93.00	97.00	99.00
hiv/aids	2056.0	0.53	0.68	0.10	0.10	0.10	0.80	1.80
gdp	2056.0	5083.28	5479.41	1.68	574.52	3055.97	7464.49	16784.35
population	2056.0	7347377.52	8598433.49	34.00	421182.25	3370017.00	11813315.38	28656282.00
thinness_10-19_years	2056.0	4.67	3.97	0.10	1.50	3.30	7.00	15.30
thinness_5-9_years	2056.0	4.69	4.01	0.10	1.50	3.35	7.10	15.50
income_composition_of_resources	2056.0	0.64	0.17	0.29	0.50	0.68	0.77	0.95
schooling	2056.0	12.08	3.12	4.90	10.30	12.30	14.20	19.50

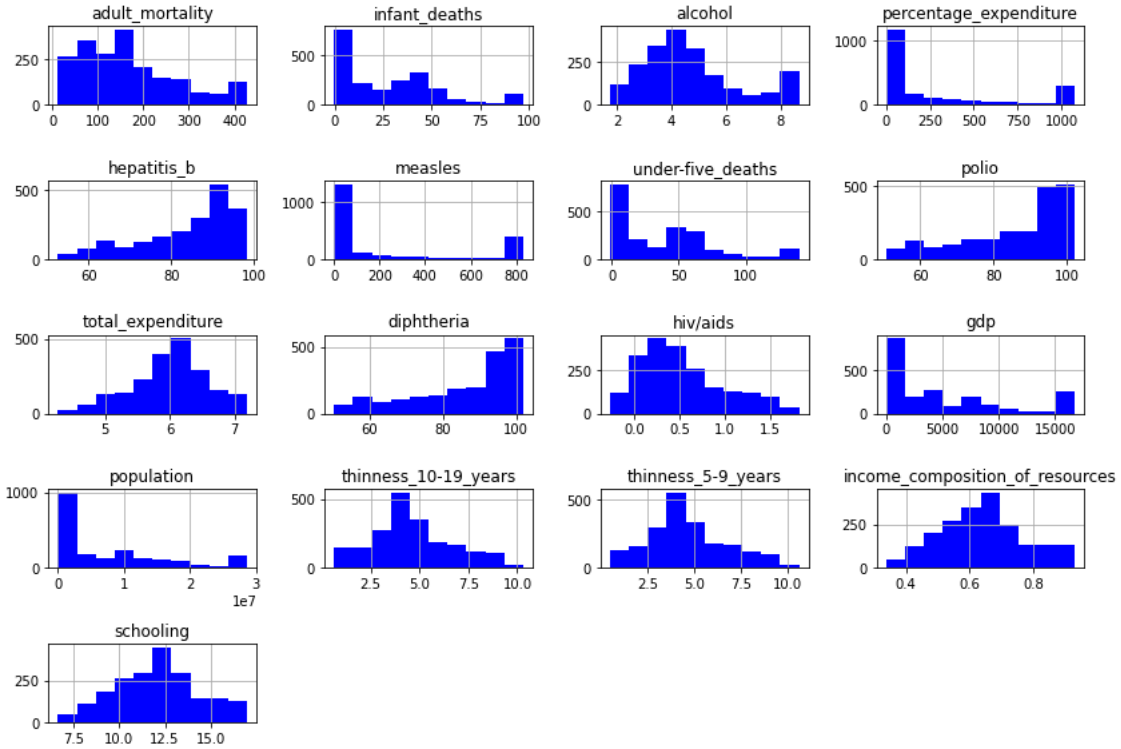
6.12.3. Correlation Matrices of Original & Synthetic Dataset



```
Text(0.5, 1.0, 'Synthetic Data Features Correlation Matrix Heatmap')
```







6.12.4. Random Forest Evaluator Performance

```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mae',
                      max_depth=6, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=60, n_jobs=-1, oob_score=True,
                      random_state=4, verbose=0, warm_start=False)
```

```
rf.oob_score_.round(2)
```

```
0.88
```

```
R2 Score of RandomForestRegressor : 0.88
Root Mean Squared Error Score of RandomForestRegressor : 3.17
```

6.12.5. XGBoost Validator

```
R2 Score of XGBRegressor : 0.91 Root Mean Squared Error Score of XGBReg
ressor : 2.76
```

6.12.6. Cross Validation Results of Evaluator and Validator

	fit_time	score_time	test_score	train_score
0	8.906651	0.103840	0.855607	0.906562
1	7.496632	0.103503	0.882802	0.916941
2	7.498439	0.103537	0.886082	0.917701
3	7.221879	0.103523	0.901223	0.922588
4	7.628481	0.103533	0.873062	0.920505

```
print('Mean Test Score of Random Forest Regressor: ', np.round(scores_rf['test_score'].mean(),2))
print('Mean Train Score of Random Forest Regressor: ', np.round(scores_rf['train_score'].mean(),2))
```

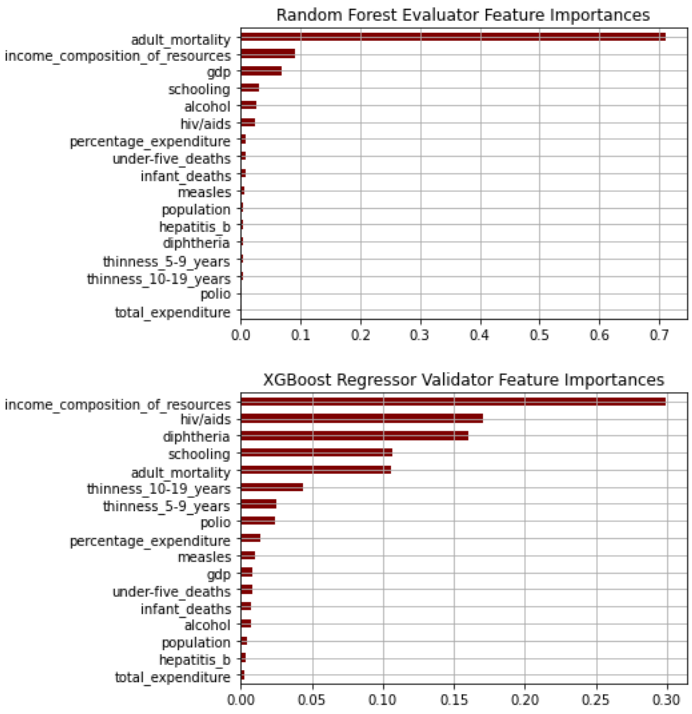
```
Mean Test Score of Random Forest Regressor:  0.88
Mean Train Score of Random Forest Regressor:  0.92
```

	fit_time	score_time	test_score	train_score
0	0.066373	0.002590	0.937020	0.950857
1	0.059190	0.002638	0.934076	0.946847
2	0.055378	0.002952	0.919495	0.949659
3	0.057373	0.002627	0.931156	0.952268
4	0.056681	0.002612	0.928419	0.949702

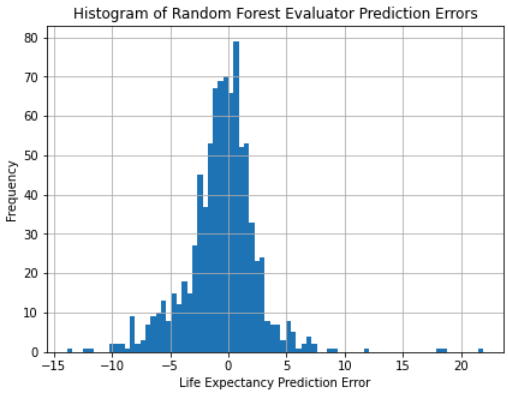
```
print('Mean Test Score of XGBoost Regressor: ', np.round(scores_xgb['test_score'].mean(),2))
print('Mean Train Score of XGBoost Regressor: ', np.round(scores_xgb['train_score'].mean(),2))
```

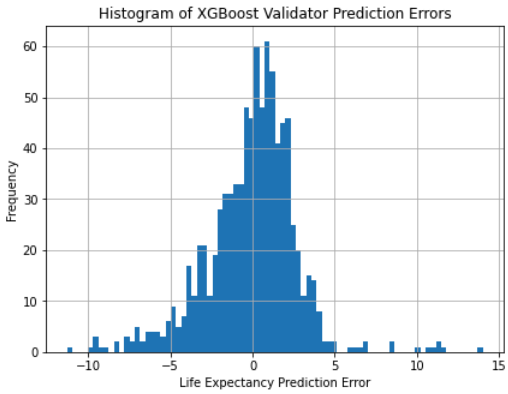
```
Mean Test Score of XGBoost Regressor:  0.93
Mean Train Score of XGBoost Regressor:  0.95
```

6.12.7. Feature Importance Ratings of Evaluator and Validator

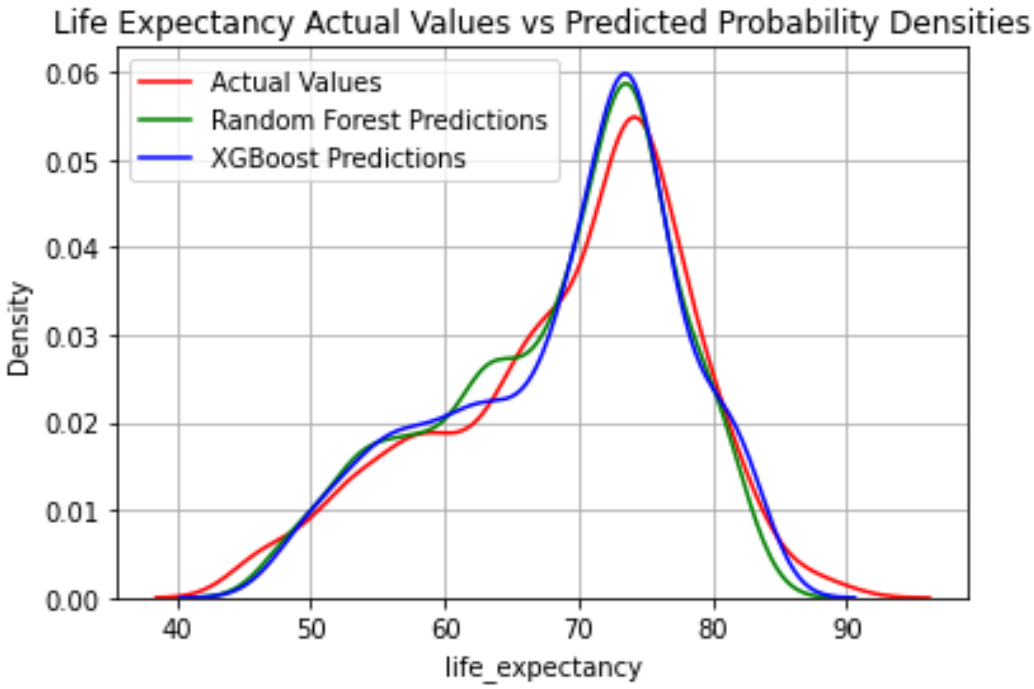


6.12.8. Model Residual Distributions





6.12.9. Life Expectancy Actual vs Model Predicted Values



6.12.10. Summary

The first step was to clean the data, this included detecting and dealing with both missing values and outliers. The variables and dataset were given a general description so that a better understanding of what the variables mean could be gathered. Then both explicit and inexplicit missing values were detected. Inexplicit missing values were values that didn't make sense for a variable given the nature of the data. There were a number of seemingly nonsensical values found given many variables' descriptions. Those inexplicit missing values were then converted to explicit missing values or nulls. Interpolation would have likely been the best method to deal with the now explicit null values (since it is time series data), but interpolation in this case would not have garnered any results. Therefore, the next best thing was done instead, imputation based on the means of all countries by year. Once missing values were sorted, the next step was detecting and dealing with outliers. Extreme value detection was done primarily by using standard box and whisker plots with a standard IQR

threshold of 1.5. Using this technique, each variable's data was winsorized on a one by one basis to eliminate outliers while limiting the loss of data. Once this step was complete, exploration of the data could be conducted.

The now clean dataset was then analyzed using univariate and bivariate techniques. One of the univariate techniques used was to inspect continuous variables using histograms in order to get an idea of their distributions. The general descriptive statistics were also found for the continuous variables. After that, categorical count plots were created to get an idea of the 'distribution' of categorical data. From that analysis it was discovered that the majority of the data fell under the 'Developing' country status. With the univariate analysis complete, it was time to move on to bivariate analysis. Bivariate analysis definitely laid most of the groundwork for understanding the relationships not only between the target variable (Life Expectancy) and the other variables, but also every variable compare to one another. The primary method used in the bivariate analysis was by the use of the correlation matrix in conjunction with the heatmap visual from the Seaborn library. This took care of the main comparisons between continuous to continuous data and was the main foundation for feature selection. But before moving on to feature engineering, some categorical variables were compared to the target variable. It was found that 'Life Expectancy' with respect to year did not garner significant enough difference to use in analysis. However, it was found that the 'Status' of a country did have a significant effect on 'Life Expectancy'. In addition to 'Life Expectancy' it also appeared to be significantly different for a number of other continuous variables. It is for this reason that new indicator variables, 'Developed' and 'Developing', were created in the next section, feature engineering.

7. Module 2: Image Synthetic data

7.1. . Introduction

Variational Autoencoders (VAEs) can be used to visualize high-dimensional data in a meaningful, lower-dimensional space. In this kernel, we go over some details about autoencoding and autoencoders, especially VAEs, before constructing and training a deep VAE on the MNIST data from the Digit Recognizer competition. We'll see how the data cluster in the lower-dimensional space according to their digit class. Plotting the test set data in this space shows where the images with unknown digit classes fall with respect to the known digit classes.

The code here borrows heavily from François Chollet's example VAE from his book Deep Learning with Python. You can find a repo of examples from the book (including the one that inspired this kernel) here on GitHub.

7.1.1. *What is autoencoding?*

Autoencoding is much like what it sounds in the sense that the input and 'output' are essentially the same. It's an algorithm for data compression where the functions for compression and decompression are *learned from the data*. It's considered more of a *semi-supervised* learning method as opposed to a truly *unsupervised* one since it's not entirely 'targetless'. Instead it learns the targets from the data itself.

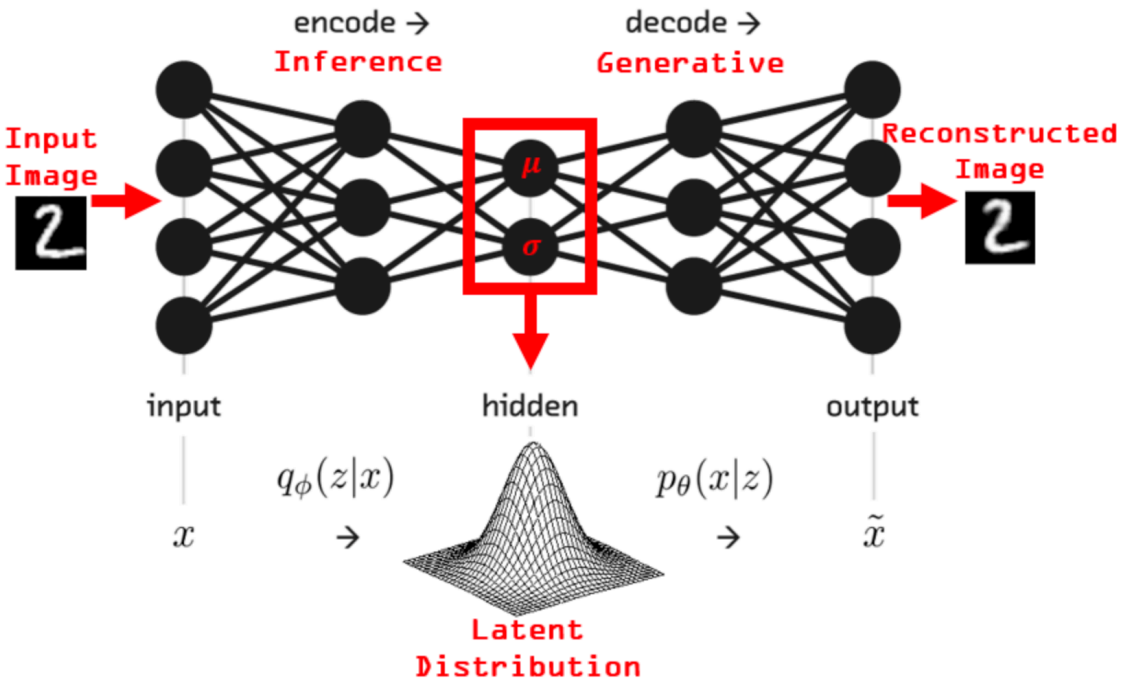
Despite all this talk of data compression, autoencoders aren't typically used for that purpose. In practice, you're much more likely to see them being used to preprocess data (as in denoising - think images but it doesn't have to be ;)) or for dimensionality reduction. In fact, the hidden layers of simple autoencoders are doing something like principal component analysis (PCA), another method traditionally used for dimensionality reduction.

7.1.2. *Autoencoders*

Generally autoencoders have three parts: an encoder, a decoder, and a 'loss' function that maps one to the other. For the simplest autoencoders - the sort that compress and then reconstruct the original inputs from the compressed representation - we can think of the 'loss' as describing the amount of information lost in the process of reconstruction. Typically when people are talking about autoencoders, they're talking about ones where the encoders and decoders are neural networks (in our case deep convnets). In training the autoencoder, we're optimizing the parameters of the neural networks to minimize the 'loss' (or distance) and we do that by stochastic gradient descent (yet another topic for another post).

7.1.3. *The Variational Variety*

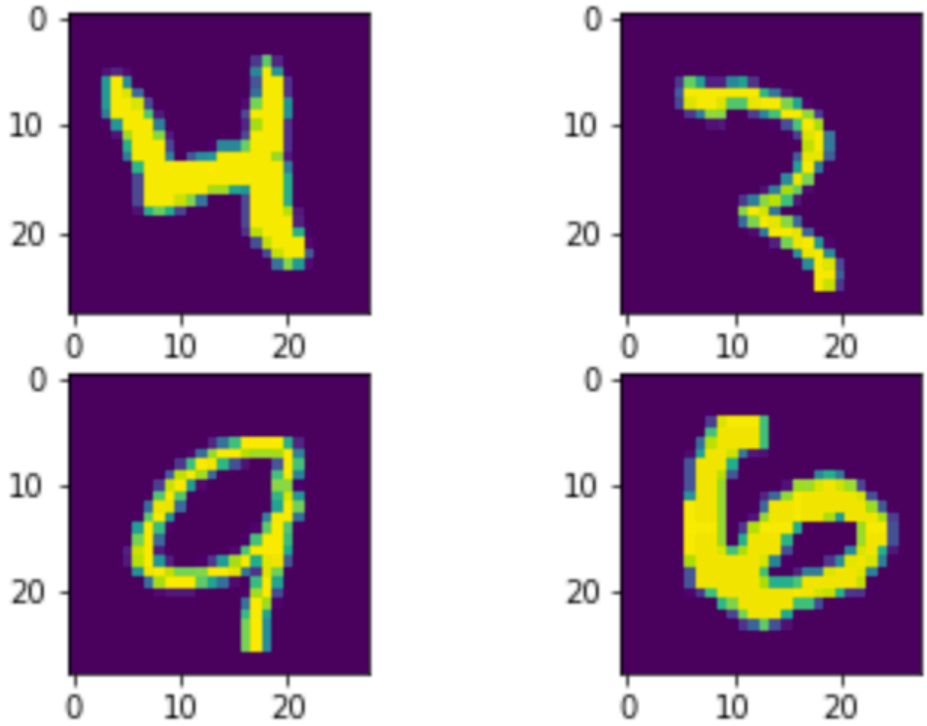
There's a bunch of different kinds of autoencoders but for this post I'm going to concentrate on one type called a *variational autoencoder*. Variational autoencoders (VAEs) don't learn to morph the data in and out of a compressed representation of itself like the 'vanilla' autoencoders I described above. Instead, they learn the parameters of the probability distribution that the data came from. These types of autoencoders have much in common with latent factor analysis (if you know something about that). The encoder and decoder learn models that are in terms of underlying, unobserved *latent* variables. It's essentially an inference model and a generative model daisy-chained together.



VAEs have received a lot of attention because of their *generative* ability (though they seem to be falling out of fashion in favor of general adversarial networks, or GANs, in that regard). Since they learn about the distribution the inputs came from, we can sample from that distribution to generate novel data. As we'll see, VAEs can also be used to cluster data in useful ways.

Show MNIST images

```
y_train[20] =  4
y_train[500] =  3
y_train[3000] =  9
y_train[9000] =  6
```



7.2. *Model* construction

7.2.1. Encoder *network*

A VAE has three basic parts:

1. An encoder that that learns the parameters (mean and variance) of the underlying latent distribution;
2. A means of sampling from that distribution; and,
3. A decoder that can turn the sample from #2 back into an image.

In this example, both the encoder and decoder networks are deep convnets. You'll notice that the encoder below has two output layers, one for the latent distribution mean (z_mu) and the other for its variance (z_log_sigma).

7.2.2. Sampling function

Next, we create a function to sample from the distribution we just learned the parameters of. `epsilon` is a tensor of small random normal values. One of the assumptions underlying a VAE like this is that our data arose from a random process and is normally distributed in the latent space.

With Keras, everything has to be in a 'layer' to compile correctly. This goes for our sampling function. The `Lambda` layer wrapper let's us do this.

7.2.3. Decoder network

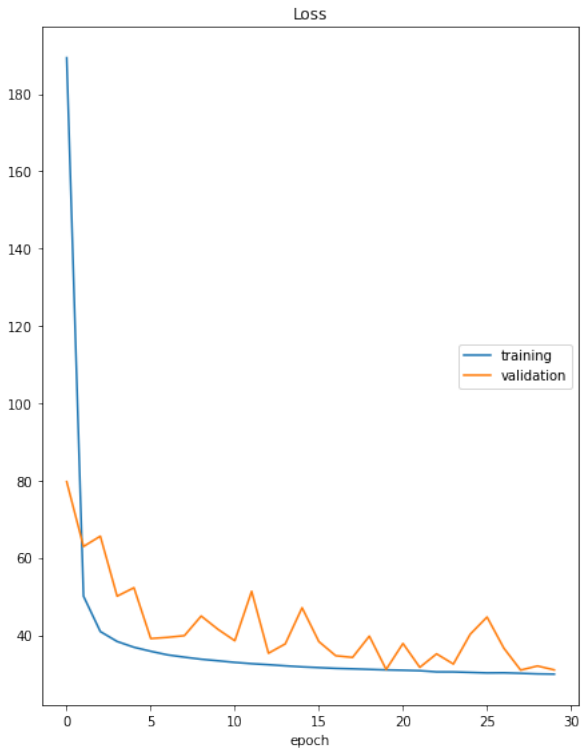
The decoder is basically the encoder in reverse.

7.2.4. Loss function

We need one more thing and that's something that will calculate the unique loss function the VAE requires. Recall that the VAE is trained using a loss function with two components:

1. 'Reconstruction loss' - This is the cross-entropy describing the errors between the decoded samples from the latent distribution and the original inputs.
2. The Kullback-Liebler divergence between the latent distribution and the prior (this acts as a sort of regularization term).

We define a custom layer class that calculates the loss.



7.2.5. Model Compilation

```
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
```

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 28, 28)]	0
reshape (Reshape)	(None, 28, 28, 1)	0
zero_padding2d (ZeroPadding2	(None, 32, 32, 1)	0
conv2d (Conv2D)	(None, 32, 32, 32)	320
activation (Activation)	(None, 32, 32, 32)	0
batch_normalization (BatchNo	(None, 32, 32, 32)	128
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
conv2d_1 (Conv2D)	(None, 16, 16, 64)	18496
activation_1 (Activation)	(None, 16, 16, 64)	0
batch_normalization_1 (Batch	(None, 16, 16, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 8, 8, 64)	0
conv2d_2 (Conv2D)	(None, 8, 8, 128)	73856
activation_2 (Activation)	(None, 8, 8, 128)	0
batch_normalization_2 (Batch	(None, 8, 8, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 4, 4, 128)	0
conv2d_3 (Conv2D)	(None, 4, 4, 128)	147584
activation_3 (Activation)	(None, 4, 4, 128)	0
batch_normalization_3 (Batch	(None, 4, 4, 128)	512
max_pooling2d_3 (MaxPooling2	(None, 2, 2, 128)	0
conv2d_4 (Conv2D)	(None, 2, 2, 128)	147584
activation_4 (Activation)	(None, 2, 2, 128)	0
batch_normalization_4 (Batch	(None, 2, 2, 128)	512
max_pooling2d_4 (MaxPooling2	(None, 1, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 2)	258

Total params: 390,018
Trainable params: 389,058
Non-trainable params: 960

Model: "sequential"

Layer (type)	Output Shape	Param #
dense_2 (Dense)	(None, 128)	384
reshape_1 (Reshape)	(None, 1, 1, 128)	0

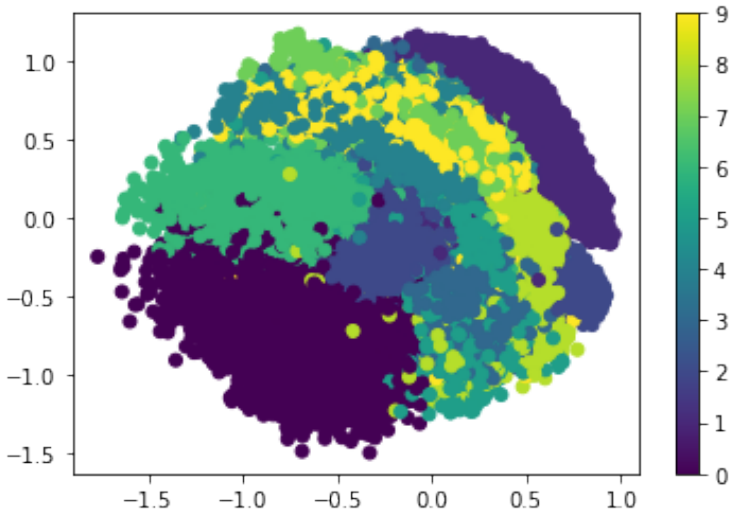
```
_________________________________________________________________
up_sampling2d (UpSampling2D) (None, 2, 2, 128)         0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 2, 2, 128)         147584
_________________________________________________________________
activation_5 (Activation)    (None, 2, 2, 128)         0
_________________________________________________________________
batch_normalization_5 (Batch (None, 2, 2, 128)         512
_________________________________________________________________
up_sampling2d_1 (UpSampling2 (None, 4, 4, 128)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 4, 4, 128)         147584
_________________________________________________________________
activation_6 (Activation)    (None, 4, 4, 128)         0
_________________________________________________________________
batch_normalization_6 (Batch (None, 4, 4, 128)         512
_________________________________________________________________
up_sampling2d_2 (UpSampling2 (None, 8, 8, 128)         0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 8, 8, 64)          73792
_________________________________________________________________
activation_7 (Activation)    (None, 8, 8, 64)          0
_________________________________________________________________
batch_normalization_7 (Batch (None, 8, 8, 64)          256
_________________________________________________________________
up_sampling2d_3 (UpSampling2 (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 16, 16, 32)        18464
_________________________________________________________________
activation_8 (Activation)    (None, 16, 16, 32)        0
_________________________________________________________________
batch_normalization_8 (Batch (None, 16, 16, 32)        128
_________________________________________________________________
up_sampling2d_4 (UpSampling2 (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 32, 32, 1)         289
_________________________________________________________________
activation_9 (Activation)    (None, 32, 32, 1)         0
_________________________________________________________________
batch_normalization_9 (Batch (None, 32, 32, 1)         4
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 32, 32, 1)         10
_________________________________________________________________
activation_10 (Activation)   (None, 32, 32, 1)         0
_________________________________________________________________
cropping2d (Cropping2D)      (None, 28, 28, 1)         0
_________________________________________________________________
reshape_2 (Reshape)          (None, 28, 28)            0
=================================================================
Total params: 389,519
Trainable params: 388,813
Non-trainable params: 706
_________________________________________________________________
Model: "functional_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 28, 28)]          0
_________________________________________________________________
reshape (Reshape)            (None, 28, 28, 1)         0
_________________________________________________________________
```

zero_padding2d (ZeroPadding2	(None, 32, 32, 1)	0
conv2d (Conv2D)	(None, 32, 32, 32)	320
activation (Activation)	(None, 32, 32, 32)	0
batch_normalization (BatchNo	(None, 32, 32, 32)	128
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
conv2d_1 (Conv2D)	(None, 16, 16, 64)	18496
activation_1 (Activation)	(None, 16, 16, 64)	0
batch_normalization_1 (Batch	(None, 16, 16, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 8, 8, 64)	0
conv2d_2 (Conv2D)	(None, 8, 8, 128)	73856
activation_2 (Activation)	(None, 8, 8, 128)	0
batch_normalization_2 (Batch	(None, 8, 8, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 4, 4, 128)	0
conv2d_3 (Conv2D)	(None, 4, 4, 128)	147584
activation_3 (Activation)	(None, 4, 4, 128)	0
batch_normalization_3 (Batch	(None, 4, 4, 128)	512
max_pooling2d_3 (MaxPooling2	(None, 2, 2, 128)	0
conv2d_4 (Conv2D)	(None, 2, 2, 128)	147584
activation_4 (Activation)	(None, 2, 2, 128)	0
batch_normalization_4 (Batch	(None, 2, 2, 128)	512
max_pooling2d_4 (MaxPooling2	(None, 1, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 2)	258
sequential (Sequential)	(None, 28, 28)	389519

Total params: 779,537
Trainable params: 777,871
Non-trainable params: 1,666

7.3. Clustering of digits in the latent space

We can make predictions on the validation set using the encoder network. This has the effect of translating the images from the 784-dimensional input space into the 2-dimensional latent space. When we color-code those translated data points according to their known digit class, we can see how the digits cluster together.

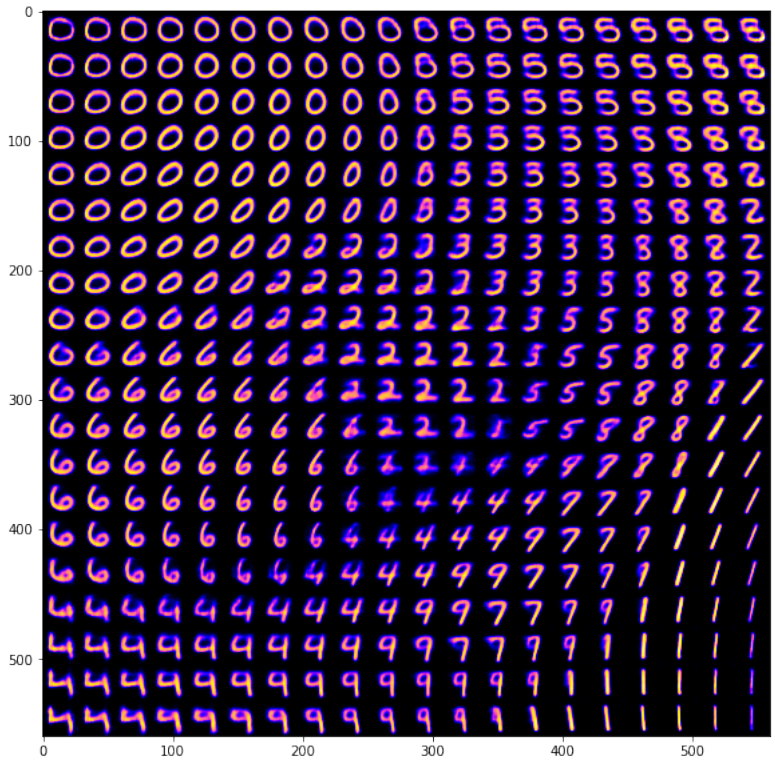


7.3.1. Reconstructing Digits

Autoencoder predictions are the compressed representations of the digits themselves.

7.3.1. Generating Digits By Sampling From Latent Space

Another fun thing we can do is to use the decoder network to take a peak at what samples from the latent space look like as we change the latent variables. What we end up with is a smoothly varying space where each digit transforms into the others as we dial the latent variables up and down.

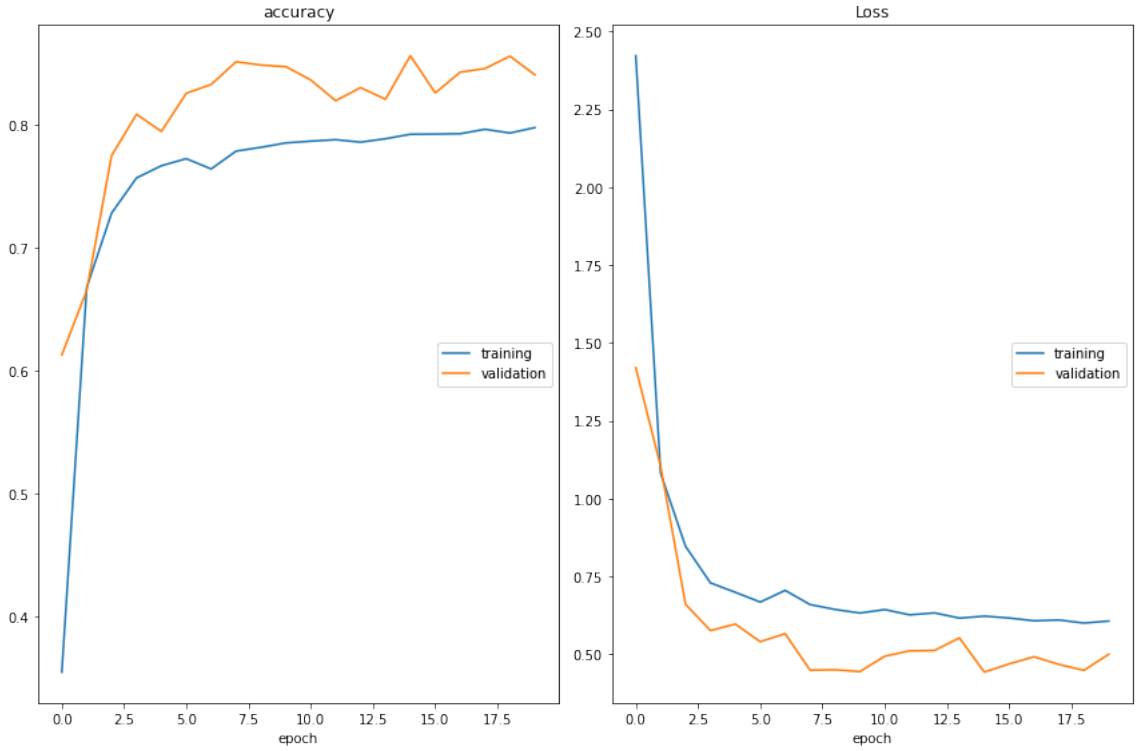


7.4. Discriminator:

7.4.1. Convolutional Neural Network Validator

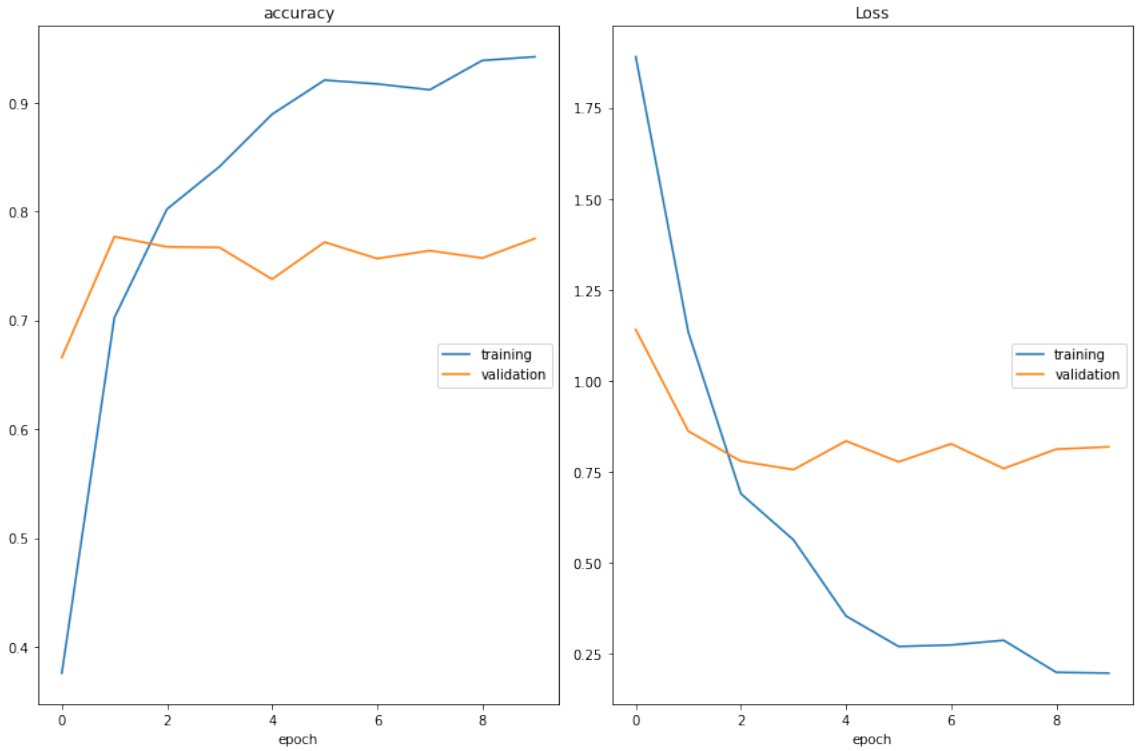
The CNN validator results can be observed with 84% accuracy as below

```
accuracy
        training                 (min:    0.355, max:    0.798, cur:
0.798)
        validation               (min:    0.613, max:    0.856, cur:
0.841)
Loss
        training                 (min:    0.599, max:    2.422, cur:
0.606)
        validation               (min:    0.442, max:    1.420, cur:
0.499)
6/6 [==============================] - 4s 598ms/step - loss: 0.6061 -
accuracy: 0.7978 - val_loss: 0.4995 - val_accuracy: 0.8407
```

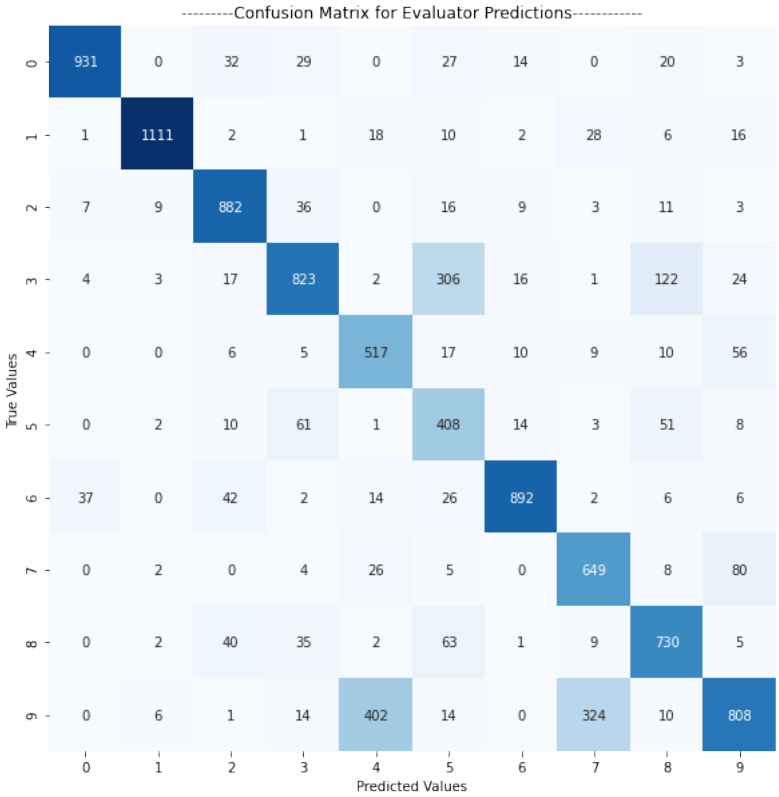


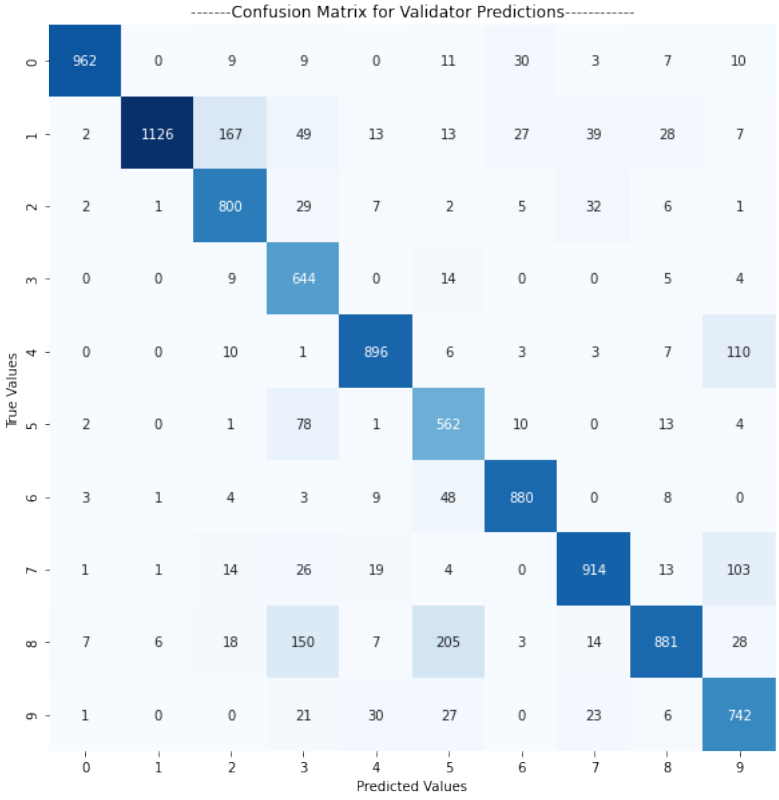
7.4.2. Convolutional Neural Network Evaluator

The Evaluator part for accuracy and loss are given below.



7.4.3. Comparing Evaluator & Validator Results






```
------------------ Evaluator Classification Report ----------------------

              precision    recall  f1-score   support
```

0	0.88	0.95	0.91	980
1	0.93	0.98	0.95	1135
2	0.90	0.85	0.88	1032
3	0.62	0.81	0.71	1010
4	0.82	0.53	0.64	982
5	0.73	0.46	0.56	892
6	0.87	0.93	0.90	958
7	0.84	0.63	0.72	1028
8	0.82	0.75	0.78	974
9	0.51	0.80	0.62	1009
accuracy			0.78	10000
macro avg	0.79	0.77	0.77	10000
weighted avg	0.80	0.78	0.77	10000

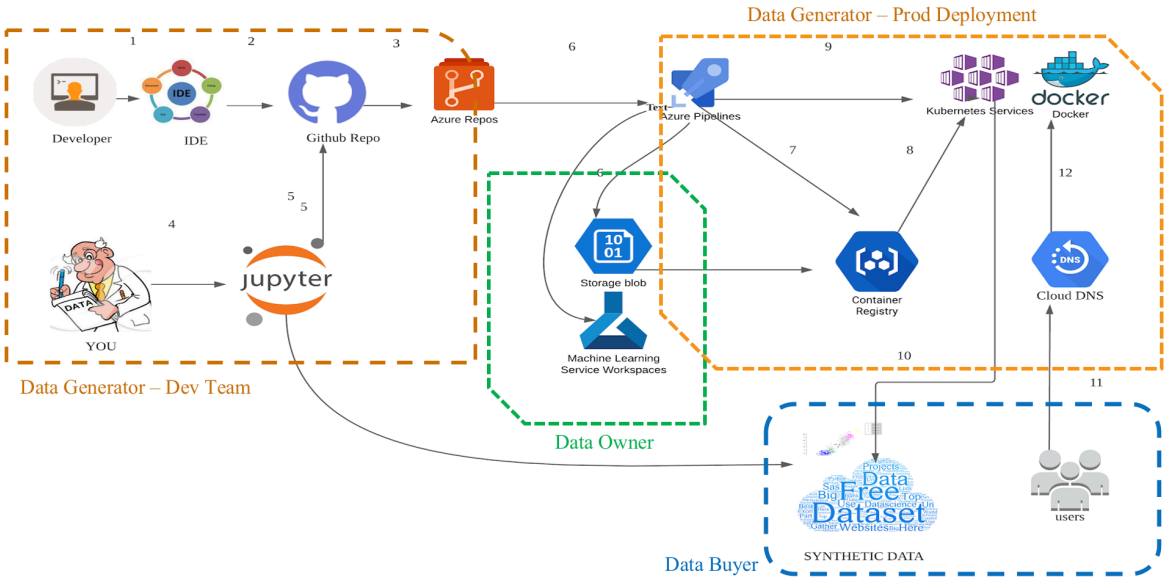
---------- Validator Classification Report ----------

	precision	recall	f1-score	support
0	0.92	0.98	0.95	980
1	0.77	0.99	0.86	1135
2	0.90	0.78	0.83	1032
3	0.95	0.64	0.76	1010
4	0.86	0.91	0.89	982
5	0.84	0.63	0.72	892
6	0.92	0.92	0.92	958
7	0.83	0.89	0.86	1028
8	0.67	0.90	0.77	974
9	0.87	0.74	0.80	1009
accuracy			0.84	10000
macro avg	0.85	0.84	0.84	10000
weighted avg	0.85	0.84	0.84	10000

8. SYSTEM DESIGN AND ARCHIITECTURE

The implementation part has been fully automated and deployable across varous cloud platforms and we chose to go ahead with Azure. The pipeline flow is shows as below. It contains 3primary section for data privacy as given.

Data Generator: The part where we generate the data
Data owner: The part where we identify the scope of data
Data Buyer: The consumer who will need data.
The same app is webhosted and will be available over:
URL http://101.127.128.81:8080



9. INDIVIDUAL CONTRIBUTION:

Individual contribution is uploaded as separate files in same directory.

10. RESEARCH AND REFERENCES

A. https://ec.europa.eu/info/law/law-topic/data-protection/data-protection-eu_en

B. https://www.linkedin.com/pulse/prospects-limitations-synthetic-data-robin-r%C3%B6hm/

C. https://www.forbes.com/sites/bernardmarr/2018/11/05/does-synthetic-data-hold-the-secret-to-artificial-intelligence/#1a63c1da42f8

D. https://www.tandfonline.com/doi/full/10.1080/2058802X.2019.1668192

E. https://github.com/keras-team/keras/blob/master/examples/variational_autoencoder_deconv.py