**Faculty of Business and Hospitality**

**2019-2020 Academic Year**

**Semester 1**

**Master of Science in Data Analytics**

****

**Programming for Data Analytics**

**Final Assignment**

**NumPy, Pandas, and Matplotlib**

**Submitted by**

**Yash Sinojia - A00268852**

**CONTENTS**

1. **Introduction………………………………………………………….. 3**
2. **Editor of Choice……………………………………………………… 4**
3. **Tutorial………………………………………………………………… 6**
   1. **Import Libraries……………………………………………….. 6**
   2. **Reading Dataset………………………………………………. 7**
   3. **Data Cleaning………………………………………………….. 8**
   4. **Summary Statistics…………………………………………. 11**
   5. **Visualizations………………………………………………… 12**
      1. **Pie Chart……………………………………………….. 12**
      2. **Regression Plot………………………………………. 13**
      3. **Histogram……………………………………………… 14**
      4. **Heat Map……………………………………………….. 16**
      5. **Violin Plot……………………………………………… 17**
4. **Reviews on Libraries……………………………………………… 18**

**4.1 NumPy………………………………………………………… 18**

**4.2 Pandas………………………………………………………… 20**

**4.3 Matplotlib……………………………………………………... 21**

**4.4 Seaborn……………………………………………………….. 22**

**4.5 Other Essential Libraries…………………………………... 23**

**4.5.1 SciPy…………………………………………………... 23**

**4.5.2 Statsmodels………………………………………….. 24**

**4.5.3 Scikit-learn…………………………………………… 24**

**4.5.4 Scrapy………………………………………………… 25**

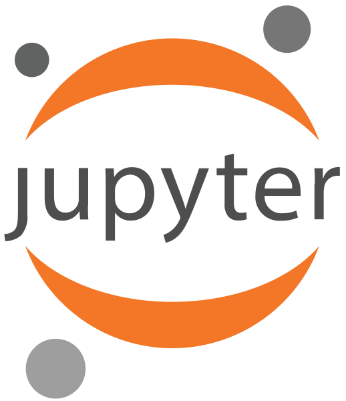
**4.5.5 TensorFlow………………………………………….. 26**

1. **References………………………………………………………….. 27**
2. **Introduction**

Python has been and will be for a long time in near future, an increasingly popular tool for data analytics. In this decade, a number of Python libraries have reached a mature developmental stage. Python users enjoy the functionalities of R, MATLAB, SAS and Stata as an alternative without sacrificing the beauty and flexibility of Python experience.

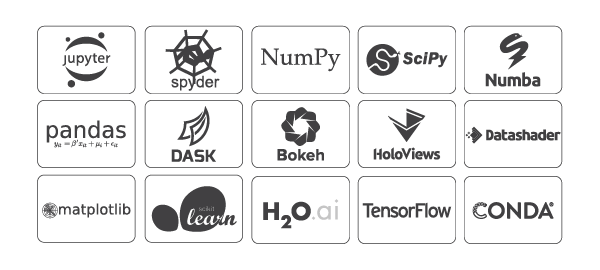
Amongst all the data analytics libraries: NumPy, Pandas and, Matplotlib are the most significant and influential. This report demonstrates the scopes of these libraries with a worked upon example tutorial that shows the step-by-step process of data analytics from data cleaning to visualization and the expressive roles these packages play in the process. After that the features and usability of these prime analytics libraries are described with some other nonetheless essential libraries to consider as per as the requirements in a field.

1. **Editor of Choice: Jupyter Notebook**

***Fig 1: Jupyter Notebook with Anaconda Distribution***

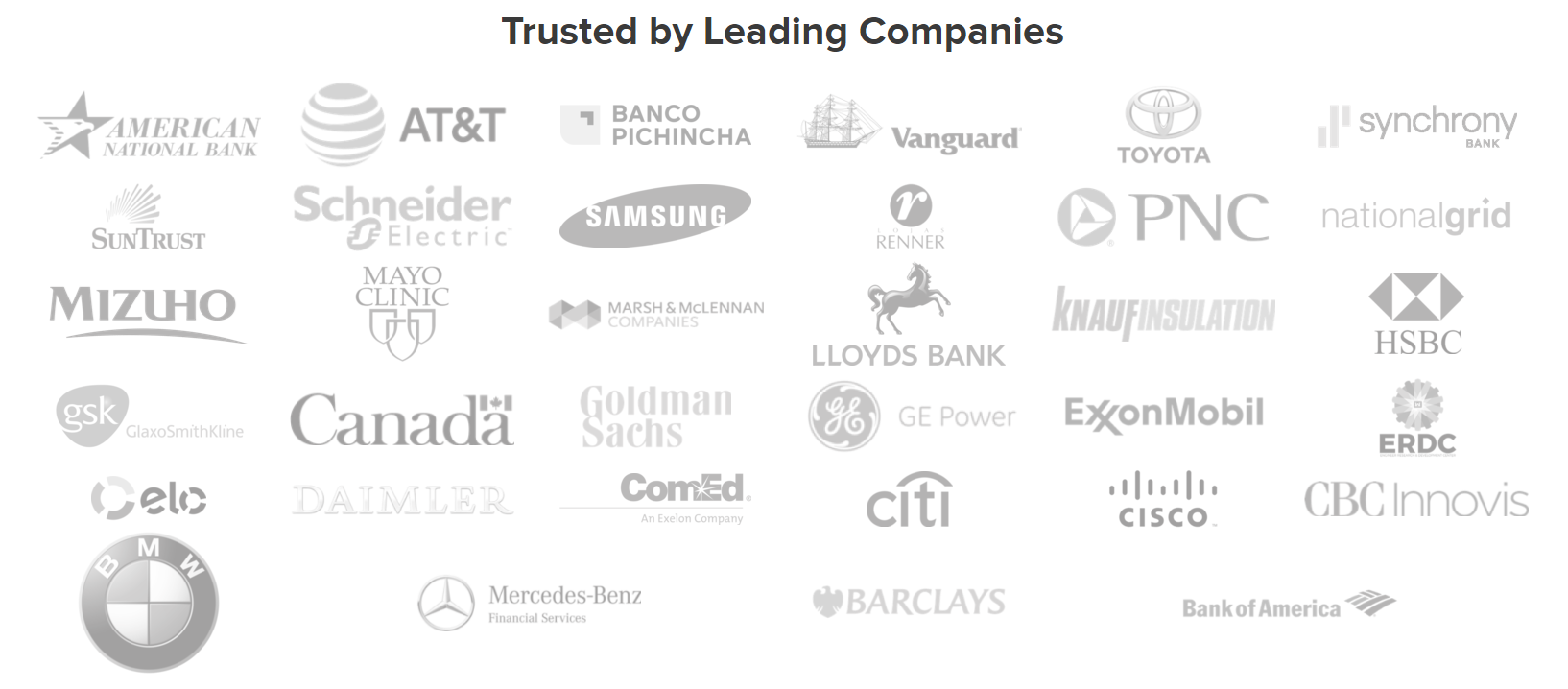
* + Preinstalled Numpy, Matlplotlib, Pandas, SciPy and all the other important python libraries and frameworks.



***Fig 2: Some Libraries and Frameworks preinstalled with Anaconda***

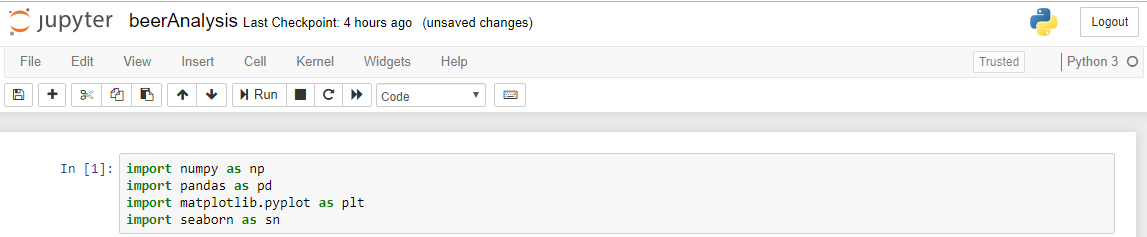
* + Easy to use live code notebook interface.
  + Web application that runs on localhost, no internet connection required.
  + Embedded sharing functionalities via email, Dropbox, Github and nbviewer.
  + Diverse options to produce outputs: HTML, images, videos, LaTeX, etc.
  + Comes with integration for big data tools such as Apache Spark, Scikit-learn, TensorFlow, etc.
  + Autosave feature
  + Markdown reporting
  + Save notebook with widget state information for static rendering, i.e. JSON format.

(*Project Jupyter | Home*, no date)



***Fig 3: MNCs using Anaconda Distributions***

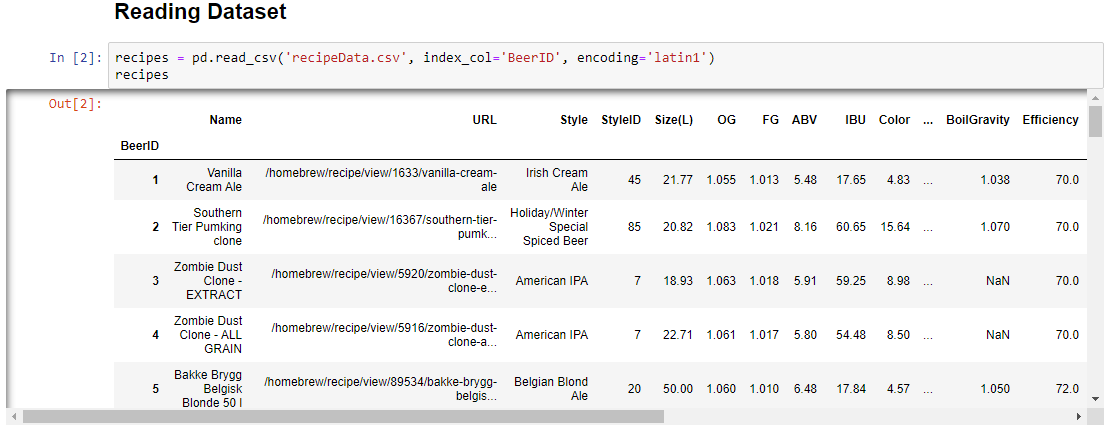
1. **Tutorial**
   1. **Import Libraries**



***Fig 4: Importing Libraries in Jupyter***

This tutorial focuses on performing an exploratory data analysis with an extensive dataset on homebrewed beer recipes found from Kaggle (*Brewer’s Friend Beer Recipes | Kaggle*, no date) using python libraries: NumPy, Pandas and Matplotlib. The analysis revolves around importing/exporting a dataset, cleaning the data to a relevant format, generating various kind of insightful visualizations from that data.

* 1. **Reading Dataset**



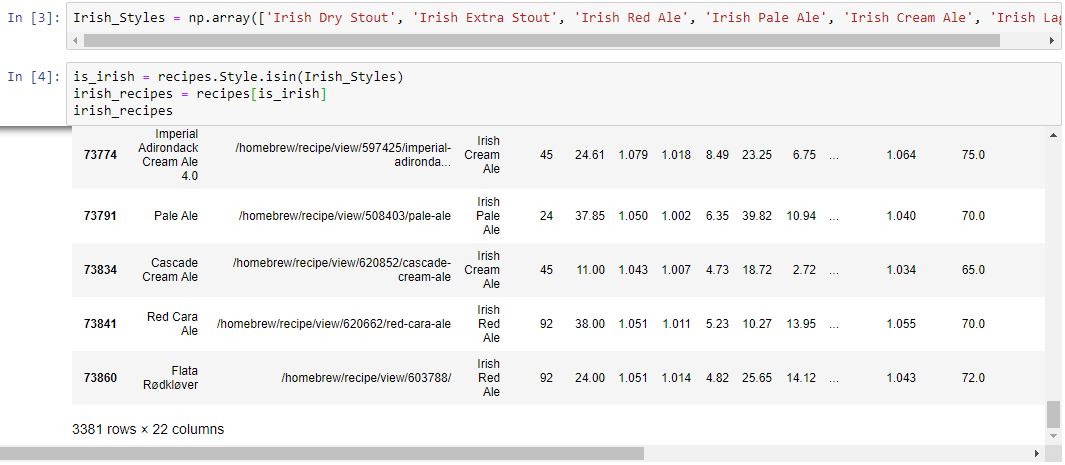


***Fig 5: Reading Dataset to a Pandas Dataframe***

The dataset has been imported from a csv file into a pandas dataframe. A pandas dataframe assumes the first row to be a header as default. An extra parameter has to be added to consider index column. This dataframe has 73861 rows and 22 columns.

Now, here we filter out the dataframe by Irish beer styles as follows:

* Irish Red Ale
* Irish Pale Ale
* Irish Cream Ale
* Irish Dry Stout
* Irish Extra Stout
* Irish Lager



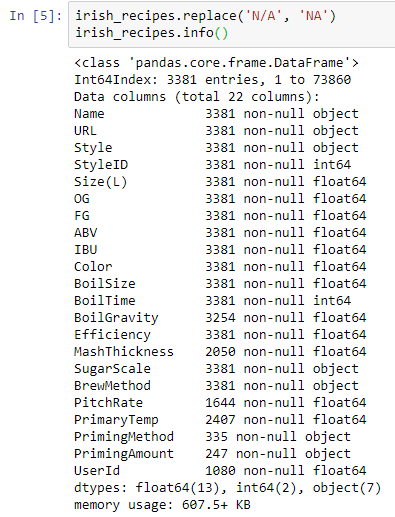
***Fig 6: Filtering out Irish Beer Styles***

Now, we have 3381 rows and 22 columns in the dataframe.

* 1. **Data Cleansing**

Here, various techniques are applied to find missing values, replacing it with sample statistics and pruning the dataset.

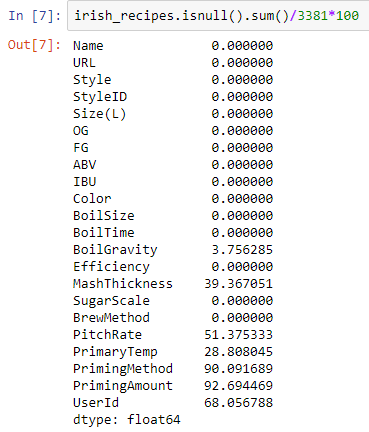
There are several values as ‘N/A’ in the data, we convert these to NA for Python to recognize it as null values.



***Fig 7: Count of non-null values in each column***

Above information describes the number of non-null values and its datatype corresponding each column in the dataframe.

Now, we gather percent of null values in each column.



***Fig 8: % of null values in each column***

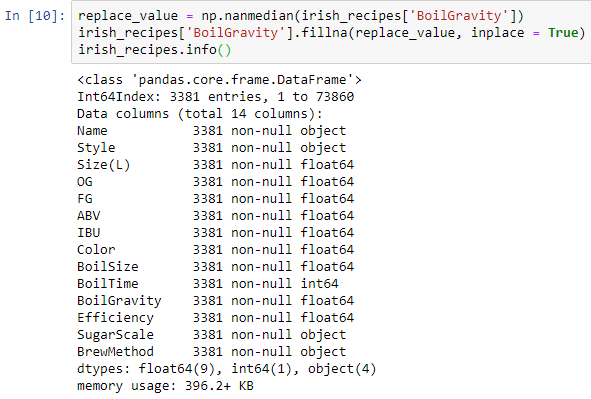
As a general statistical rule, any column having greater than 25% of its values as null can become too much biased and bad for the analysis. So, we drop all the columns with >25% null values. Hence columns: ‘MashThickness’, ‘PitchRate’, ‘PrimaryTemp’, ‘PrimingMethod’, ‘PrimingAmount’ and ‘UserId’ are dropped.



***Fig 9: Dropping columns from the data***

Here, we also drop ‘URL’ and ‘StyleID’ additionally as they are extraneous for analysis.

We still have the column ‘BoilGravity’ with < 4% of its data as null values. As ‘BoilGravity’ consists of outliers. (displayed as a significant difference between its mean and median in summary statistics) We replace the null values with the median of all the other non-null values.



***Fig 10: No null values in the dataframe***

Now, the dataset does not have any null values in it.

There is a column in dataset called ‘SugarScale’ that describes the unit of columns: ‘OG’, ‘FG’ and ‘BoilGravity’ as either Specific Gravity or Plato.



***Fig 11: Count of Plato units***

There are only 73 units in the dataset as Plato which are affecting the values in ‘OG’, ‘FG’ and ‘BoilGravity’ columns. So, it’s better to get rid of it.



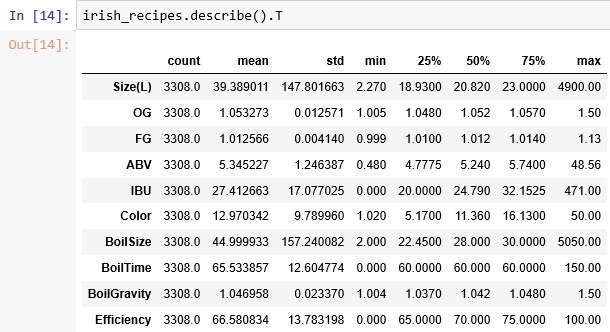
***Fig 12: Extracting rows with Specific Gravity units***

* 1. **Summary Statistics**



***Fig 13: Dimension of dataframe***

There are 3308 rows and 14 columns after cleaning the dataframe. The summary statistics is as follows:



***Fig 14: Summary Statistics of dataframe***

Exporting this summary to a csv file.



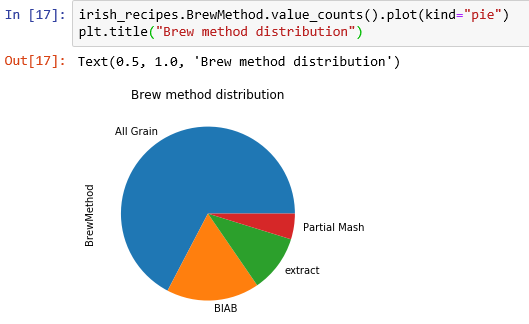
***Fig 15: Saving to .csv file with NumPy***

And storing the cleaned dataframe to a csv file as ‘irishRecipes.csv’.



***Fig 16: Saving to .csv file with Pandas***

* 1. **Visualizations**
     1. **Pie Chart**



***Fig 17: Brew Methods Pie Chart***

More than 60% of the recipes are brewed with all grain method.

* + 1. **Regreesion Plot**

We adjust the upper bound by taking BoilSize < 1000 and then we plot a regression to identify if there is a relationship between boil size and boil time.

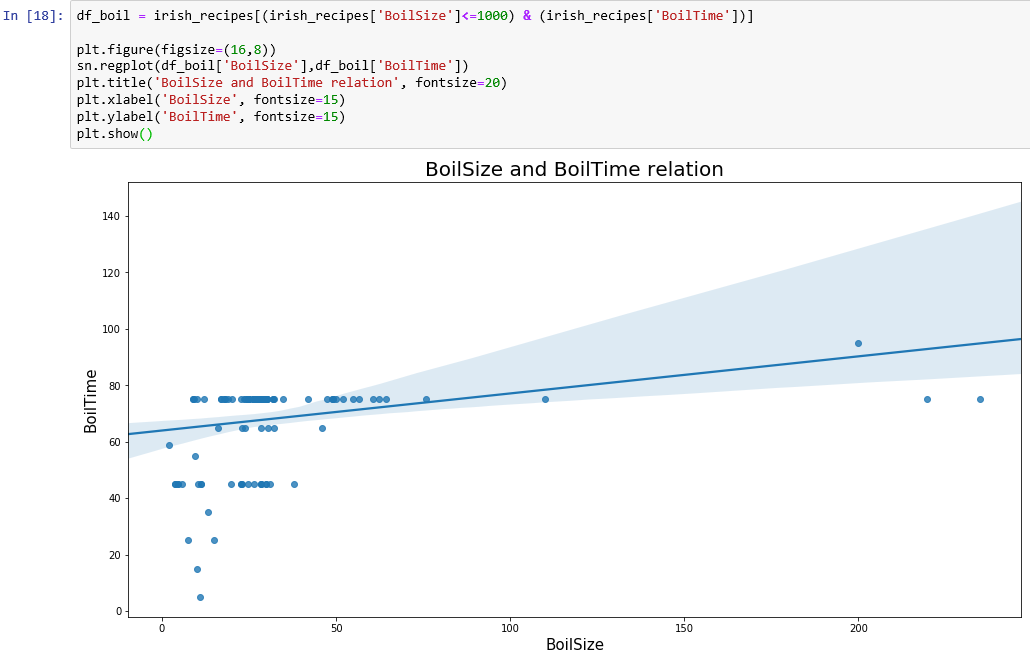
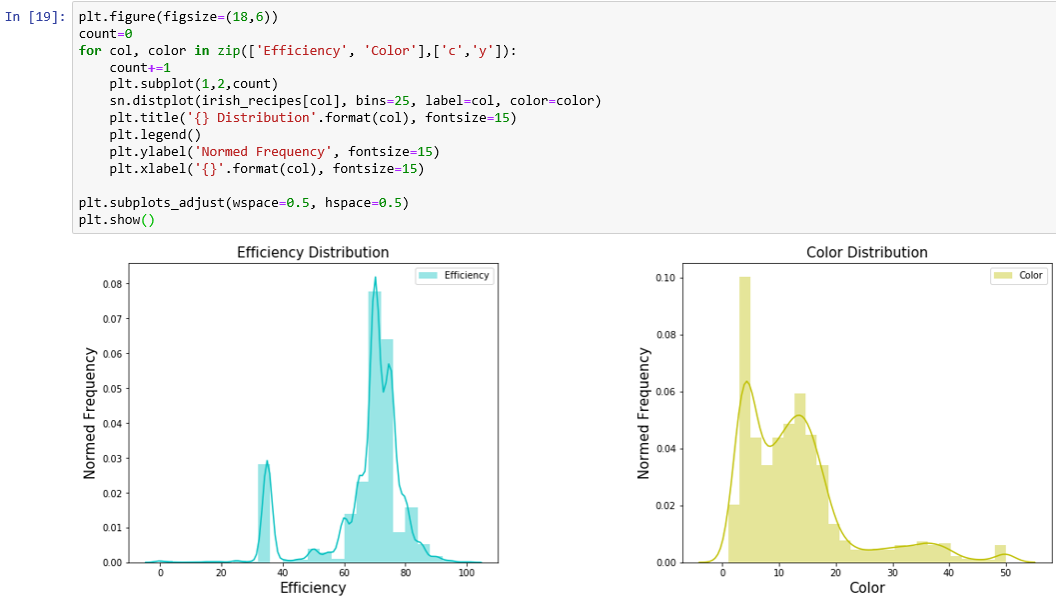
***Fig 18: BoilTime vs. BoilSize Regression plot***

Figure shows there is bit of linear correlation between boil size and boil time, when boil size < 1000 while the boil time is nearly constant with a gradual steep.

* + 1. **Histogram**

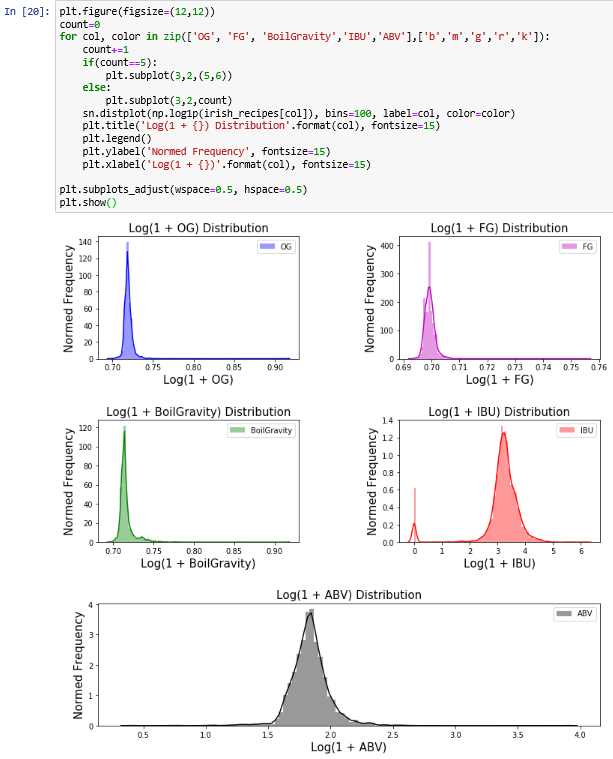
Histograms of ‘Efficiency’ and ‘Color’ with normed frequency to identify with their nature of distribution.



***Fig 19: Normal Histograms for Efficiency and Color***

Efficiency is right skewed and Color is left skewed.

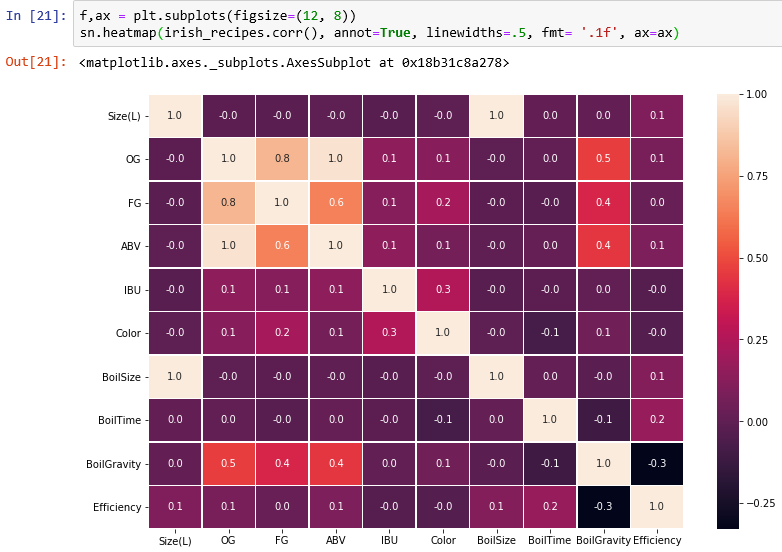
Histograms of ‘OG’, ‘FG’, ‘BoilGravity’, ‘IBU’ and ‘ABV’ columns with logarithmic scale to get a better visualization of data with outliers.



***Fig 20: Logarithmic Histograms for OG, FG, BoilGravity, IBU and ABV***

OG, FG and BoilGravity has a lot of outliers on the right, while IBU and ABV have nearly normalized distribution.

* + 1. **Heat Map**

To find which columns in the dataset are correlated to each other, we calculate Pearson correlation coefficient (1 = perfect positive, 0 = no correlation, -1 = perfect negative) between every two columns and visualize it on a heat map.

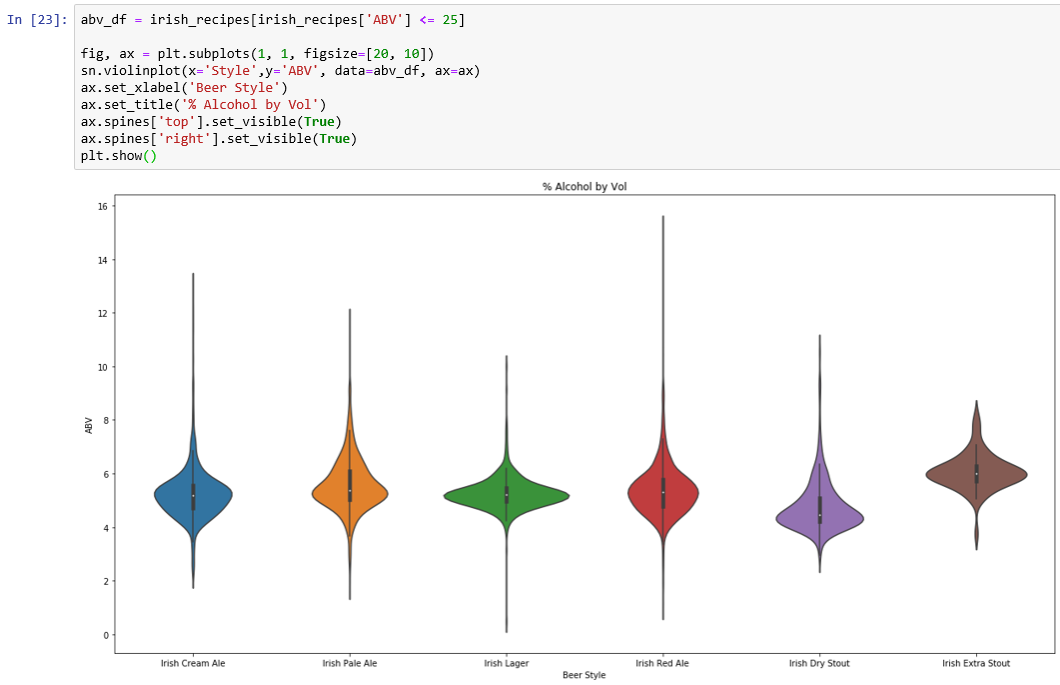
***Fig 21: Heatmap for correlations among columns***

There are high direct relations of ABV with OG, OG with FG, ABV with FG. As scientifically, ABV can be estimated as ABV = (OG - FG) \* 131.25 (*Original Gravity (OG) & Final Gravity (FG) Calculator - Straight 2 The Pint*, no date), which is backed up by the heatmap.

* + 1. **Violin Plot**

Violin plots are better alternatives for boxplot to describe the spread of data

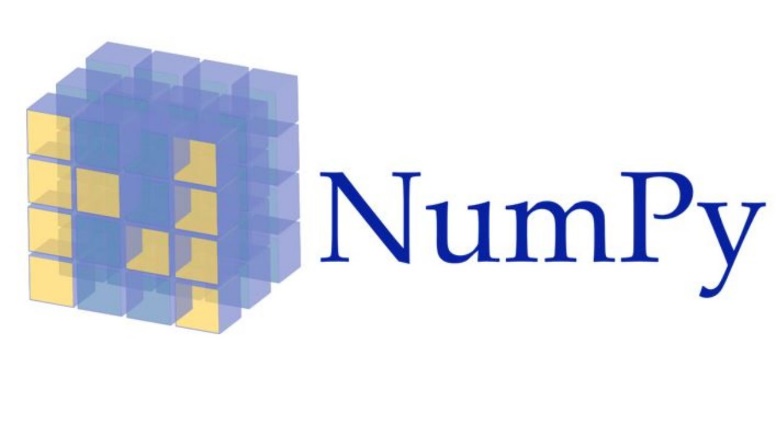
points inside the interquartile range and outliers outside it.



***Fig 22: Violin plots on %ABV***

ABV in Irish Red Ale has the greatest number of outliers, while Irish Extra Stout has the least. On average, the alcohol content seems the most in Irish Extra Stout. The data is most spread close to median in Irish Lager, whilst its least spread around median in Irish Red Ale.

1. **Reviews on Libraries**
   1. **NumPy**



***Fig 23: NumPy***

1. NumPy is a numerical python library.
2. Supports large, multi-dimensional arrays and matrix data structures.
3. Renders a large collection of high-level mathematical functions to operate on these arrays and matrices.
4. Why NumPy instead of Lists?

There is nothing in NumPy which is not there in either lists or other data structures, but

* NumPy is much faster
  + Uses fixed type which by default only utilizes 4 bytes for an integer (Int32). Also customizable to 2 bytes (Int16) and 1 byte (Int8).
  + While a list utilizes 4 bytes for size, 8 bytes for reference count, 8 bytes for object type and another 8 bytes for object value which totals to 28 bytes for an integer.
  + NumPy operates only on homogenous values, so no object type overhead when iterating through objects.
  + A NumPy array utilizes less bytes of memory and hence its faster to read.
  + A list occupies non-contiguous blocks of memory to store its data.
  + While, Numpy occupies contiguous blocks of memory which assists CPU’s SIMD vector processing units for efficient cache utilization and hence easier retrieval of data.
  + The dimensions of a NumPy array can be manipulated during runtime as long as multiplicity factor produces the same number of products. e.g. reshape() function.
  + NumPy renders API for n-dimensional arrays filled with 0s, 1s or any other values, whose data type can be changed with dtype() function.
  + NumPy can implement plethora of mathematical functions on arrays in the fields of trigonometry, algebra and statistics.

1. NumPy also contains random number generators.

(*A hitchhiker guide to python NumPy Arrays - Towards Data Science*, no date)

1. Usability

* A standalone library for mathematical operations.
* However, it is less flexible for I/O functionalities.
  + Tried to store the summary statistics in a csv file as described in the tutorial. But header has to be mentioned separately and there is no direct functionality to include index in data. Alternatively, a lot simpler in Pandas.
* NumPy act as a backend to extend Pandas and Matplotlib libraries.
  1. **Pandas**



***Fig 24: Pandas***

* + 1. Major python library, well known for data manipulation and analysis.
       - Provides data manipulation and data control like SQL queries.
    2. From basic to advanced building blocks for doing practical, real world data analytics.
    3. Provides various data structures and operations for manipulating tables and time series.
* Flexible to convert a CSV, TSV or SQL database to a python object of rows and columns called a series or dataframe with integrated indexing by ultrafast HDF5 format.
  + 1. Reshaping and subsetting of datasets.
    2. Insertion, deletion, merging and joining
    3. Time-series functionalities
    4. Label based slicing and filtering like Excel.
    5. Hierarchical labelling of axes.
    6. Mimics major functionalities of R, an alternative.
    7. Pandas is fast on one feature operation at a time, as low-level algorithmic bits being already tweaked in CPython code, but extrapolation may sacrifice the performance.

(*Package overview — pandas 0.25.3 documentation*, no date)

* + 1. Usability
       - Has to be used with NumPy for efficient mathematical operations.
       - Best for data manipulations, no better alternatives in Python.
       - Reduces lines of code significantly as compared to basic python codes.
       - Most useful in cleaning a dataset. i.e. finding missing values, replacing or removing it.
  1. **Matplotlib**



***Fig 25: Matplotlib***

Plotting and visualization library for Python.

Renders an object-oriented API framework for embedding plots into program using GUI tools i.e. GTK+, Qt, wxPython, Tkinter, etc.

Mimics MATLAB in its visualization and graphics functionality, a free and open-source alternative.

* + - * 1. Matplotlib has a module as Pyplot that provides interface like MATLAB.

SciPy makes use of Matplotlib

Allows access to enormous data in easily interpretable infographics.

Supports several kinds of plots:

Line plot

Histogram

Scatter plot

Line plot

Polar plot

Contour plot

3D plot

Image plot

(*Data Visualization in Python: Matplotlib vs Seaborn*, no date)

Usability

The infographics are more visually emphatic if used with seaborn.

Coding becomes a bit complicated for multi-categorical data, have to mention parameters for each category separately.

More customization to be done to avoid overlapping if a lot of labels in legends.

Can be used with seaborn to avoid complicacies in plotting code.

* 1. **Seaborn**



***Fig 26: Seaborn***

Data Visualization library based on Matplotlib.

Provides a higher-level interface for making attractive and informative graphs.

Adds an extra layer of visual beauty on Matplotlib plots.

E.g. mapping a colour to a variable or using faceting.

Provides several optionality for themes, color patterns, axes calibration, margin customizations, gridline setups, adding various annotations of shapes and text.

(*Seaborn – The Python Graph Gallery*, no date)

Usability

Makes Matplotlib easier to use by adding default as well as customization settings and reduces the lines of code.

Can’t be used without Matplotlib Pyplot module.

* 1. **Other Essential Libraries**

**4.5.1 SciPy**



***Fig 27: SciPy***

Scientific computing and technical computing library for Python.

Mimics computing abilities of MATLAB and GNU Octave, a free and open source alternative.

Contains modules for linear algebra, optimization, Fourier transformations, integration, differentiation, signal processing and image processing.

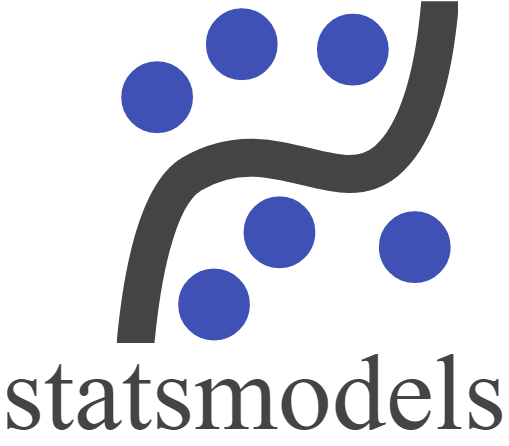
SciPy builds upon NumPy array object stack.

Has to be used with Matplotlib to visualize its computing results.

Basic data structure is matrix array.

(*SciPy - Wikipedia*, no date)

**4.5.2 StatsModels**



***Fig 28: Statsmodels***

1. Statistical python library with major focus on statistical models and statistical tests.

Descriptive Statistcs

Inferential Statistics

Hypothesis testing

Estimation

1. A complement to SciPy for statistical computing.

(*statsmodels · PyPI*, no date)

**4.5.3 Scikit-learn**



***Fig 29: Scikit-learn***

1. Python library for machine learning and data mining toolkits.

Classification

Regression

Clustering

Dimensionality reduction

Model selection

Preprocessing

Cross-validation

1. Designed to interpolate with NumPy and SciPy

(*scikit-learn: machine learning in Python — scikit-learn 0.21.3 documentation*, no date)

**4.5.4 Scrapy**



***Fig 30: Scrapy***

1. Web-scraping and web-crawling library for Python.

Used to generate a dataset from web by scratch.

Any accessible information from a website can be converted to a dataset.

Can also extract data from APIs

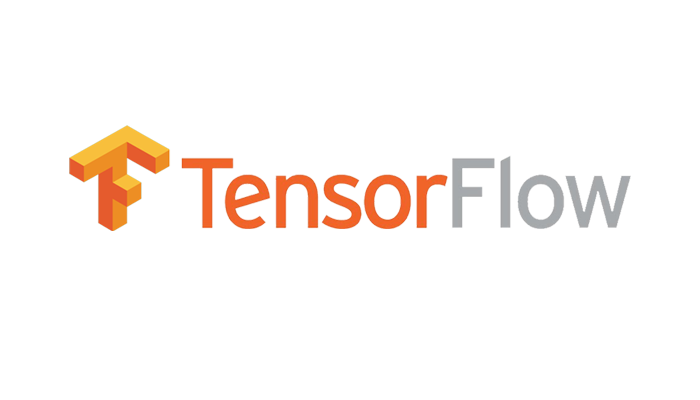
1. Architecture is built around “spiders”: self-contained crawlers embedded with instructions.
2. Usability

Can be used to build up a raw dataset to be further analyzed by Pandas.

An independent framework

(*Scrapy - Wikipedia*, no date)

**4.5.5 TensorFlow**



***Fig 31: TensorFlow***

* 1. End-to-end, open source, deep learning and artificial intelligence platform for Python.
  2. Developed by Google.
  3. Large applications for neural networks.
  4. Allows dataflow and differentiable programming across a range of tasks.
  5. It provides a backend support for Graphics Processing Unit to render large computational power for complex programs.

(*TensorFlow - Introduction - Tutorialspoint*, no date)

1. **References**

*A hitchhiker guide to python NumPy Arrays - Towards Data Science* (no date). Available at: https://towardsdatascience.com/a-hitchhiker-guide-to-python-numpy-arrays-9358de570121 (Accessed: 22 November 2019).

*Brewer’s Friend Beer Recipes | Kaggle* (no date). Available at: https://www.kaggle.com/jtrofe/beer-recipes (Accessed: 17 November 2019).

*Data Visualization in Python: Matplotlib vs Seaborn* (no date). Available at: https://www.kdnuggets.com/2019/04/data-visualization-python-matplotlib-seaborn.html (Accessed: 22 November 2019).

*Original Gravity (OG) & Final Gravity (FG) Calculator - Straight 2 The Pint* (no date). Available at: https://straighttothepint.com/original-gravity-og-final-gravity-fg-calculator/ (Accessed: 22 November 2019).

*Package overview — pandas 0.25.3 documentation* (no date). Available at: https://pandas.pydata.org/pandas-docs/stable/getting\_started/overview.html (Accessed: 22 November 2019).

*Project Jupyter | Home* (no date). Available at: https://jupyter.org/index.html (Accessed: 22 November 2019).

*scikit-learn: machine learning in Python — scikit-learn 0.21.3 documentation* (no date). Available at: https://scikit-learn.org/stable/index.html (Accessed: 22 November 2019).

*SciPy - Wikipedia* (no date). Available at: https://en.wikipedia.org/wiki/SciPy (Accessed: 22 November 2019).

*Scrapy - Wikipedia* (no date). Available at: https://en.wikipedia.org/wiki/Scrapy (Accessed: 22 November 2019).

*Seaborn – The Python Graph Gallery* (no date). Available at: https://python-graph-gallery.com/seaborn/ (Accessed: 22 November 2019).

*statsmodels · PyPI* (no date). Available at: https://pypi.org/project/statsmodels/ (Accessed: 22 November 2019).

*TensorFlow - Introduction - Tutorialspoint* (no date). Available at: https://www.tutorialspoint.com/tensorflow/tensorflow\_introduction.htm (Accessed: 22 November 2019).