

Data Science Programming

10204281

Section (5)

Submitted To:

Dr. Raneem Qaddoura

Submitted By:

Nasrullah Rami Alhaj Hamad

* 1. **Types of Data Structure:**

Data structures are a way of designing the code and storing data so they can be accessed and used later in the program,

The data structures in Python are either built-in, such as Tuples, Lists, Dictionaries, and sets, or user-defined, like Stacks, Trees, Graphs, and HashMap. (Narang, 2023)

The built-in data structures are divided into two main types, Mutable and Immutable structures, which means that the data inside the structure can be changeable by either removing or adding, or changing the elements of the structure, or not.

* **Lists:** Lists are structures that vary in size. The great thing about them is that they are mutable data structures; the data inside they can be defined by Square Brackets [ ], or by using the list function which is frequently used when programming a code.

Lists have many ways to remove, add, or change the values within, which are the following:

1. Append: Which adds values to the end of the list
2. Insert: Adds values to a certain index in the list
3. Pop: Removes and returns a particular element by inserting a specific index
4. Remove: Removes without returning a certain element by inserting the value name
5. Can be concatenated by + or by ‘extend’, if a value is in the list by ‘in’ or ‘not in.’
6. Sort: lists can be sorted according to an option chosen by the user.
7. Slice: can choose multiple values in a list in a sequence using ‘: ‘

* **Tuples:** Tuples are structures that are immutable and cannot be changed which is a drawback, have a fixed length, can include any data type, and can be declared by Parenthesis ( ).

To change the tuples’ values when they are immutable, the objects can be stored in a list, and then the tuple built-in function can be used to return the list as a tuple.

Objects can be counted using the ‘count’ method and can be sorted using the ‘sorted’ method according to a sorting methodology of the user’s choosing.

Tuples are used to present an ordered selection of elements without worrying about them being changed, which is why it is commonly used in the programming world.

* **Dictionaries:** dictionaries are a way of structuring data by using keys and value pairs, which can be modified, deleted, or retrieved which is an advantage.

Dictionaries can be created using the curly brackets { }, and each key and value are separated by a comma from other keys and values. And they are unordered structures which is a drawback of them.

Existing keys can be checked by using keys(), and values by values(). And by using the ‘in’ and ‘not in’ operations. Elements can be added by applying the dictionary name to a list with the key which will equal the value of the added key. (Ruhi, 2022)

To remove either the key or values, we can use ‘del’ to delete the values, or ‘pop’ to delete the key and return the value.

Every data structure can be turned into a dictionary by using the built-in function ‘dict’. Dictionaries can be merged into one dictionary using the ‘update’ function.

* **Sets:** are an unordered structure that includes several unique elements. Sets are mutable so elements can be removed or added upon.

Sets can be declared and created by curly brackets { }, or by the built-in function set().

Elements can be added to sets with the ‘add()’ function for a single element and the ‘update()’ function for multiple elements.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ##########  ########## | Lists | Tuples | Dictionaries | Sets |
| Constructor | [ ] | ( ) | { } | { } |
| Mutable or no | Yes | No | Yes | Yes |
| Ordered or no | Yes | Yes | No | No |
| Example on it | Avg = [95 , 79] | Grow = (‘g’, ‘t’) | Date = {‘July’ = 7, ‘August’ = 8} | Trees = {‘apples, ‘palm’} |

An advantage of sets is that they can be used to use operations like union, intersection, difference, and symmetric difference.

**1.2: Common libraries:**

Libraries in Python allow the programmer to do wider operations in simpler and easier ways. They can be imported by the 'import' commands, and then they can be used by associating them with a variable. Some of the libraries Python offers are:

1. **NumPy:** This stands for Numerical Python, mainly used to work with arrays, scientific operations, and matrices. The arrays the NumPy library uses are called the ndarray object, which can differ in dimensions. It is usually shaped by rows and columns. The arrays can be merged, transposed, sliced, indexed, and reshaped. The values inside can be added, removed, and sorted.
2. **Pandas:** The pandas library is mainly used for data cleaning and analyzing situations by using data structures and data manipulation tools. The data structures pandas provide are series and dataframes.

Series are a one-dimensional version of a dataframe containing a sequence of numbers and are connected to a column of indexes. Many operations can be done to the pandas series such as concatenating them, removing values, adding them, and sorting them too.

A single dataframe contains a table of data, with an ordered number of columns, the data types can either be numerical or strings or Booleans. Like the pandas series, the dataframes can also be operated by a lot of different functions available by the library for the programmer.

1. **Scikit Learn:** The sci-kit learn library is built on NumPy and SciPy and matplotlib, and it’s a library that includes many simple and productive tools to predict data and analyze it. The library contains many efficient ways to train and test data for model training and machine learning, examples of models that are available from the sci-kit learn library are: Decision Tree, GNB, and Linear Regression.
2. **Theano:** Theano is a library that optimizes the compiler to be efficient for evaluating mathematical and matrix calculations, it was built on NumPy, and it is a stable optimizer that solves unstable statements using other stable statements. Theano is mainly used for problem-solving and machine-learning operations. (Anon. 2022)
3. **Keras:** This library focuses on the creation of functions and layers and building them to be effectively used by programmers, Keras is built on a human-based approach that makes machine learning possible and easily accessible for anyone to use. (Anon. 2022)

**1.3: Plotting and visualization libraries**

Visualization libraries are used to visualize data in an effective proper way, for the need of passing on your findings, after having done several data cleaning and analyzing steps, plotting the data measures or evaluations on charts or graphs is necessary, to build a relationship between the data and patterns.

1. Matplotlib: is one of the main libraries that are used for visualizing and plotting the relation between data. It is related to the NumPy library so static, animated, and interactive visualizations can be created.

The following statement is used to import the library: “ import matplotlib.pyplot as plt ” There are a lot of functions that are needed to create a proper visualization chart or plot, such as:

* plt.show(): which is used to show the figures in the program or code.
* plt.grid(): It is used to display a grid inside the figure.
* plt.savefig(): this function saves the plot.
* plt.legend(): this function is used to add a legend to the axes.
* plt.title(): this is used to create a title for the figure.

There are three visualization plot types, each has a different declaration method. One is the box plot, the second is the bar plot, and the third is the radar chart.

1. GGplot: This library is another popular library in Python that is used for data visualization, it uses a map to print and draw the patterns of the data, with many additions like shapes, aesthetic colors, and objects. This library is associated with the pandas library to store the data in dataframes, or series. Its documentation is easy and simple to follow.

**2.1 Programming languages and tools:**

I used a programming language called Python, which is one of the most efficient languages when operating with data science techniques, from cleaning to visualizing plots. Before, programs I wrote were run by the CMD. Now I use the google colab notebook to write and run my programs, google colab allowed me to use google servers to code. Because my device was weak, google colab gave me access to many libraries that were compatible to use with my device, the environment google colab provides to code is very efficient, and very easy and simple to use, with the libraries available to use, the codes I program can become very efficient with high performance.

Anaconda for python is a free environment that allows users to write codes and execute them properly in the Python language, it was created by continuum.io, which is a python specified company, Anaconda is one of the most famous ways of programming codes using Python since it is worked with by millions of people worldwide.

Anaconda uses the package deployment method which is widely liked by a majority of people. And it comes with several libraries ready to be used for huge projects by programmers. (Ellis, 2023)

**2.2: Load Data and Prepare Data (Preprocessing):**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Column/step Name** | **Description** | **Justification** |
|  | Age Bracket | Filling Null values with 0 | Because the age can’t be 0, and 0 is the default. |
|  | Close Date | Filling Null values with  1-1-2022 | Because this date is defined as the default date. |
|  | Open Group (1) | If it was null with the close group, it will be filled with unknown | Because the escalation flag column will have ‘No’ as a value, which indicates the case wasn’t sent to the groups |
|  | Open Group (2) | If the close group column wasn’t null, it will be filled with the mode | Because this way there would be corresponding close groups |
|  | Close Group (1) | If it was null with the close group, it will be filled with unknown | Because the escalation flag column will have ‘No’ as a value, which indicates the case wasn’t sent to the groups |
|  | Close Group (2) | If the open group column wasn’t null, it will be filled with the mode | Because this way there would be corresponding open groups |
|  | Offer Name | The missing values were filled with the mode. | Because the most existing value exceeded half of the values |
|  | Customer Group | The missing values were filled with the mode. | Because the most existing value exceeded half of the values |
|  | Open User | Missing values were filled with Anonymous | Because the user would have decided not to write their name for the case |
|  | Close User | Missing values were filled with Anonymous | Because the user would have decided not to write their name for the case |
|  | Resolution | Deleted the column | Because the null values exceed 50% |
|  | Resolution Description | Deleted the column | Because the null values exceed 50% |
|  | Case Desc | Deleted the column | Because the null values exceed 50% |
|  | Escalated Group | Deleted the column | Because the null values exceed 50% |
|  | Case ID | Deleted the column | Because it wouldn’t affect the data, and it isn’t the information that models should be trained with |
|  | Callback Mechanism | Deleted the column | Because the null values exceed 50% |

**2.3: Approaches:**

|  |  |  |
| --- | --- | --- |
| **Approach no.** | **Name** | **Description** |
|  | Gaussian Naïve Bayes | This classifier model includes algorithms that are based on applying the Bayes theorem with the naïve assumption of independence between each feature. When I used it, it was implemented while in default state. |
|  | K-nearest Neighbors | It’s a classification model that is used to determine how similar data is from each other, based on the distance between the new data and all other data, the smaller the distance the more it is similar. I used this model when the n\_neighbor value was 3. |
|  | Decision Tree | It can be both classification and regression and it works as a tree form, the datasets break into smaller data while being connected to the previous data. Decision trees can handle both numerical and categorical data. The decision tree was used in default mode. |
|  | Logistic Regression | Despite its name, it is a classifier that works by probabilities describing the possible outcomes of an iteration that are modeled correctly using a logistic function. The logistic regression model used had max\_iterations equal to 1500. |

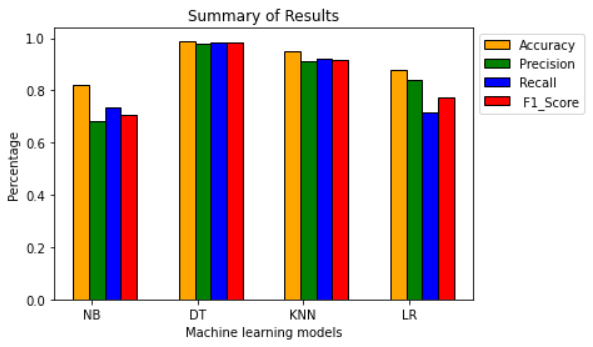
**3.1: Comparing the Different Models:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approach no.** | **Approach Name** |  | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
|  | **GNB** |  | 82.1 | 68.8 | 73.5 | 71.1 |
|  | **KNN** |  | 95.2 | 92.9 | 91.1 | 92.0 |
|  | **DT** |  | 98.7 | 98.6 | 97.2 | 97.9 |
|  | **LR** |  | 86.7 | 83.1 | 69.7 | 75.8 |

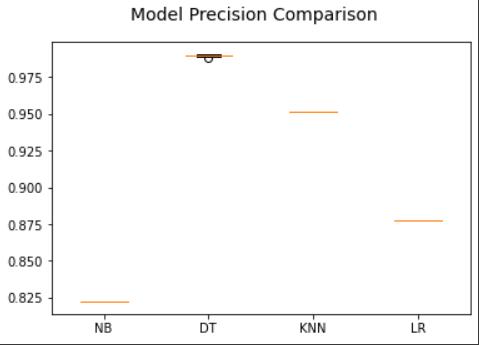
**3.2 Charts:**

The charts used for plotting and visualizing the data patterns with different models were imported with the library Matplotlib, which is a library commonly used for plotting the data. The three plotting types used were Box Plot, Bar Chart, and Radar Chart.

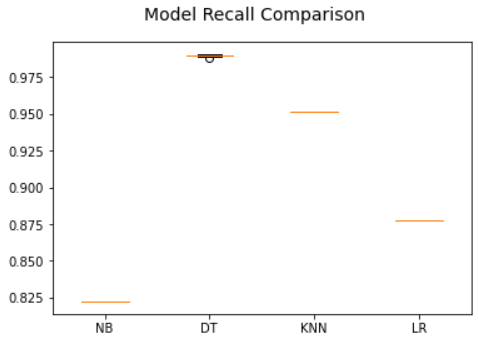
1. **Bar Chart:** The Bar Chart used contained all the different models, with the evaluation measures for each model, on a bar type of plotting.



1. **Box Plots:** Four Box plots were used in plotting the models with one plot featuring one evaluation measure. One for accuracy, one for precision, one for recall, and one for the F1-score.

Chart, box and whisker chart

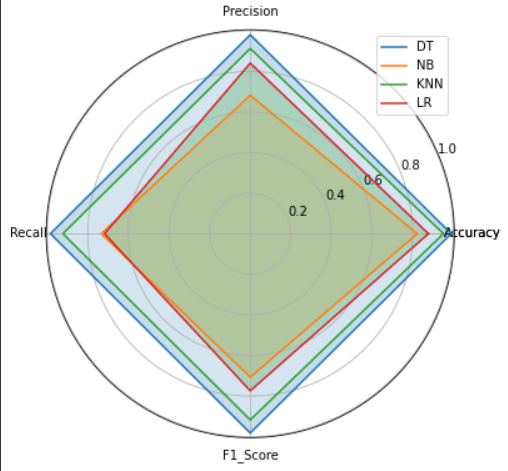
Description automatically generated

 Accuracy Precision

Chart, box and whisker chart

Description automatically generated

Recall F1-Score

1. **Radar Charts:** the radar charts are used to distribute the models on all the evaluation measures in a circular shape. The radar chart used contained all of the models and the measures of each model.

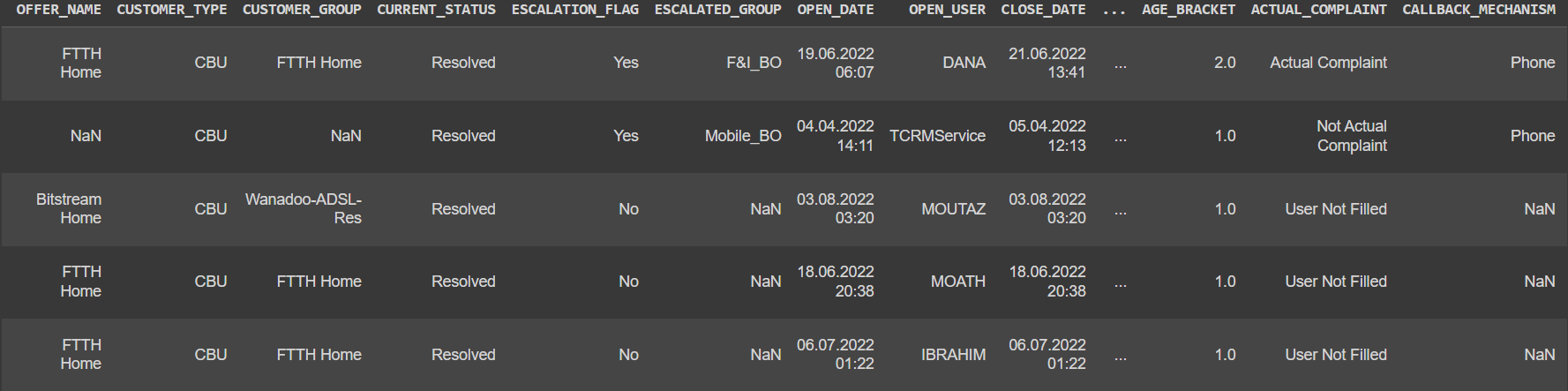
**3.3: Analyze of the results:**

The results from each model differed in the evaluation measures, the model with the highest score in all the measures, accuracy, precision, recall, and f1-score, was the Decision Tree. Following with the Logistic regression, which was close to the decision tree results, had the second-best results in all the models. The K-nearest Neighbors model was the 3rd highest model with the given measures, and the model with the lowest performances was the Naïve Bayes model.

**4.1: The choice of data structures:**

The data structures used in the code I programmed, were built on an effective way of thinking,

After I read the CSV file, I declared a variable that will create a dataframe structure that displays the dataset in a row and column shape. Having 10415 rows and 22 columns in total.

The head() function displays the first five rows of the dataframes.

Then when I started with the preprocessing session, lists were used to contain and check each column of the dataset inside a list, whether for filling in missing values or deleting the columns.

Text

Description automatically generated

Graphical user interface

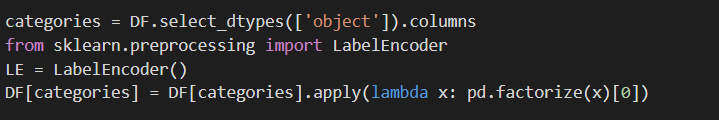
Description automatically generated with medium confidence

Then I used functions to check how the data is described by the data type and other performance measures. Using the info() and describe(), which give information about the dataframes.

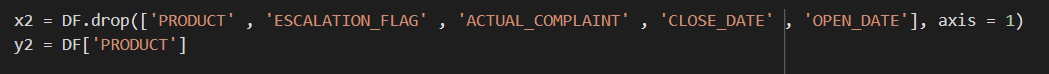




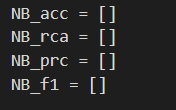
I used the variable that I stored the object type columns in using lists, when I transformed the categorical into numerical.



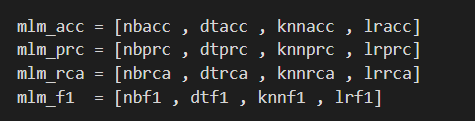
After that, I split the data using the Train\_test\_split function that I imported using the sci-kit learn library. I stored all the columns that I dropped inside a variable called ‘x2’ using a list that contained all the features, I stored the label in a list in a variable called ‘y2’.



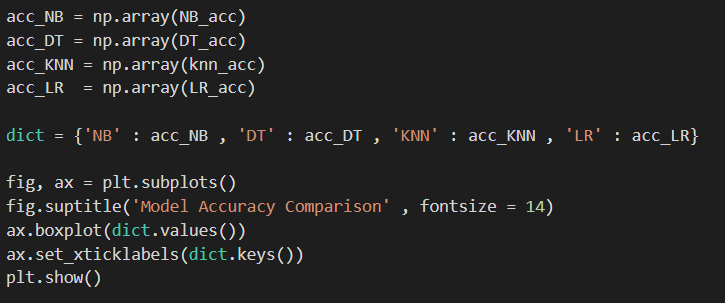
Then when I implemented the models, for each model, I used lists to store every evaluation measure possible, inside the array variable.



The Bar plot included four different types of NumPy arrays, each including one different array of measures, that were then put on the chart, as bars, for the four different models.

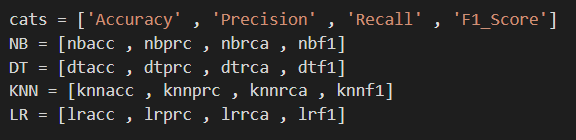


The four Box Plots included the variables that stored model names when they were associated with one evaluation measure, the names of the x-axis labels were stored in a dictionary structure as keys, and each of the variables above as values.



The Radar Chart had each model with its evaluation measures stored in numpy arrays, numbering four arrays in total. When each array had the first index in it appended, it was used inside lists.

Text

Description automatically generated

I used lists because they are ordered and the elements inside them can be changed and removed. I used the dictionaries to store the names of the models with the NumPy array variable names because they use the key and value feature which saves a lot of time and provides better performance. Dataframes were used to represent the data inside the excel sheet that was read in the beginning, using a formula of rows and columns.

**4.2: Selection of the appropriate libraries:**

I used the NumPy library to include the evaluation measures in arrays because each of the models was activated 30 times, and I used the concat function to merge the NumPy array holding the names of the measures in the Radar Chart. The NumPy library is generally known for its vast uses with arrays and matrices, which is why it was essential to use inside the program.

The Pandas library holds data structures such as series and dataframes. I used the dataframe structure to display all the data contained in the excel sheet, on a scale of rows and columns. And the preprocessing steps included methods of handling missing values. Like fillna and dropna using loops and conditions.

The sci-kit learn library is used to the data training methods, either model training or machine learning, I used four different models of the models that the library provides, and I used the Train\_test\_split function to split the data into training data fed to the model and testing data for the model to predict the same pattern as this part of data. And then I imported functions from the sci-kit learn library to calculate the four evaluation measures of the models used.

I used the visualization and plotting library known as matplotlib, which provides a selection of charts and plotting visuals, I used this library to display the data predicted by the models, and the evaluation measures, in three separate types of plotting. Bar chart, Box plots, and Radar Chart. I used the library's different functions to make the plotting more suitable and compatible with the data, such as coloring the bars and charts. And the label function to display labels to the different axis of the charts.

**4.3: The effectiveness of different models:**

In my first preprocessing stage. I used the GNB model in its default state without additional parameters, and I had good performance in it. With measures going between 75% – 85%. And I kept changing the preprocessing steps above to reach a certain level of performance. The GNB model measures kept changing for the better with measures between 80% – 90% and for the worse with measures between 70% – 80%. But at the last preprocessing stage, it reached its highest performance yet, going between 85% - 90%, but it was my worst-performing model out of the four.

When I used the DT model, it gave me the highest performance of all the models, it was used in its default stage without any additional statements inside the parenthesis. When I first trained it, it gave me measures between 90% – 99%, and when I changed the preprocessing above, its evaluation measures went even higher than before, with measures between 95% – 99%. It remained the best model out of the four models in performance level.

The KNN model was used with the K value being equal to ‘3’, and I kept it at 3 without changing it. At first, this model gave me the worst performance levels out of the four models, because It was based on different preprocessing steps from the way it was intended to be, so the measures were between 50% – 70%, then I changed the preprocessing steps once, and its evaluation measures went to be between 60% - 75%, which was better than before, but it was still my worst model with performance yet. Then I changed the preprocessing again, by dropping both the date columns, and by that, the KNN model performance went to a new level of performance, measuring between 85% - 95% which became my 2nd highest level of performance.

The Logistic regression at first was stable, it was in its default state, and produced its evaluation measures which were set in the middle of the four models, 80% – 90%. Then I changed its max iterations to 1500, which didn’t change the evaluation measures of the model, but fixed an error for me. After the preprocessing went to its second stage, the logistic regression performance stayed the same. In the last stage of the preprocessing, the model gave me its best performance, which was between 85% - 95%, It was the third-best performance I have gotten from the other models.

**4.4: Recommendations:**

After training four models to predict the data for the company, it was shown by the visualizations done to display the predicted data that the Decision Tree had the best performance in all the evaluation measures out of all the other models, so I would recommend the company to use this model for any data predictions in the future because it gives more accurate predictions.

I would recommend the company display its data in a better way, using pandas’ data frames or by using other data displaying libraries, and visualize the data so it can have a more accurate way of deciding the predictions.

I would recommend the company use a better case-handling system that would have fewer missing values in the future, so predicting data doesn’t perform low or becomes impossible.

**References…**

1. Narang, M. (2023) Top 10 python libraries for Data Visualization, KnowledgeHut. Knowledgehut. Available at: <https://www.knowledgehut.com/blog/business-intelligence-and-visualization/python-data-visualization-libraries> (Accessed: January 23, 2023).
2. Anon. (2022) Top 9 python libraries for Machine Learning in 2023 upGrad blog. Available at: https://www.upgrad.com/blog/top-python-libraries-for-machine-learning/ (Accessed: January 23, 2023).
3. Ruhi, A. (2022) Python Data Structures: Comparisons and Operations, Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/03/popular-python-data-structures-comparison-operations/ (Accessed: January 23, 2023).
4. Ellis, D. (2023) What is Anaconda for python &amp; why should you learn it?, HubSpot Blog. HubSpot. Available at: https://blog.hubspot.com/website/anaconda-python (Accessed: January 24, 2023).