

DISEASES PREDICTION BY DOCTOR TONY.

COURSEWORK_3, DATA SCIENCE

Tony, YIT19488399

UNIVERSITY OF ROEHAMPTON, 28 April 2022

Table of contents.

- + Cover page
- + Table of contents
- + Summary of data process to meet the goal
- + Description of the technique for data analysis
- + Reflection and arguments
- + References
- + Screenshots and pictures

A brief summary of how you processed/pre-processed Dataset for serving your purpose/goal.

The first block of code is based on the Naïve Bayes function applying a model using a Categorical Naïve Bayes implementation from the “SKlearn” library. “CategoricalNB()”. The dataset used has only qualitative attributes. The dataset in the data frame is split into X containing values for the predictors and y containing the label values. Only 3 symptoms are used, Symptom_1, Symptom_2 and Symptom_3 because these columns have no “NaN” values (blanks in the dataset). All the rows are used (4920). After many tests, this is the best election for better accuracy of the model.

Since all the values are qualitative/categorical, it is necessary to encode them into discrete, numerical values. “LabelEncoder” function can only process 1d arrays. So, each column is put into a 1d array and then encoded. Afterwards, aggregation is necessary for the encoded column values.

Train-test split on the dataset is selected to be 68% for the test set size. After the model is trained, 32% of the data is for the test to find the predictive output.

The “accuracy_score” function checks the accuracy between predicted and actual values of the label. In this study, 41% of accuracy is obtained. Furthermore, it is possible to predict the outcome of unseen data points to see which disease the patient has. But with the accuracy and after checking and verifying the output to predict diseases, I start to implement the second approach of the Naïve Bayes technique.

The second block of code is based on the Naïve Bayes technique as well. The first column “Disease” is the target variable that suggests different types of diseases. “SKlearn” is the Python library used to create a model and make predictions. “SKLearn” requires all features to be numerical arrays. The use of “LabelEncoder” converts categorical data into a number ranging from 0 to n-1 where n is the different values for a column. After variables and target is defined, the model

should be created by splitting data into train and test. Train dataset is to build the model and validate the model with the test dataset and see the performance of the model. From academic theory, 70% of the total data was used to train the model and 30 % after testing the model. All the seventeen symptoms were used and all the rows (4920 in total).

The model is using the Gaussian approach to the Naïve Bayes theorem. “GaussianNB()”. Now, it is possible to make predictions on the test features. The performance of the model is high, using “accuracy_score”. 87% is a very good score and the model can be considered reliable.

Finally, making predictions about which disease has the patient based on symptoms input is possible. This should be done with numbers and the predicted disease will be a numerical output too. [number]. But with the tables patients and doctors can find the human language with minimum effort. Three examples are included at the end of this second approach to predicting diseases.

The third block of code started with the intention to make some nice plots and pictures of the dataset. Meanwhile familiarising myself again (third time) with data, I found from Kaggle [6] a code with 100% accuracy. 20% and 80 % to train the model were studied and both give 100 % accuracy. But this is not a Naïve Bayes technique. This third approach is using the concept of trees. A decision tree algorithm. “RandomForestClassifier()”, and is used in a supervised dataset as Naïve Bayes.

Description of the Implementation of the model that you have done using your chosen technique of data analysis method from coursework 2 (You don't copy the code in this section but just provide the screenshots of your important code blocks with a proper title(s)).

The fundamental difference between both approaches is the model. The first one is “CategoricalNB()”, while the second one is “GaussianNB()”. There are more approaches like Multinomial, Complement and Bernoulli [1].

While implementing code block one, I observed that is better to include only three symptoms or fewer. When a fourth symptom is included the accuracy of the model fall drastically. I think this is because of the beginning of blanks in the original dataset. And only 41% of accuracy is obtained. I decided to use 68% of the data to train the model. But sometimes, using less training data, better accuracy was obtained. In summary, the first program works better with fewer symptoms input and with less training data. Therefore, I decided to implement a second program!

In the code block two [2, 3], much higher accuracy is obtained in the test of the model, 87%. The model is trained following the academic recommendation, around 60% to 80% for training. And the patient can have more symptoms, until 17 at maximum. Like in the previous block of code, the dataset is translated into numbers...

The last block of code is the most accurate but is not a Naïve Bayes approach. I found specifically this dataset to avoid plagiarism.

I think this study could be implemented with the KNN algorithm much easier and simple by translating to 1s and 0s depending on if a patient has or no symptoms in the whole dataset. There are more approaches... The reason is that I decided to use the Naïve Bayes technique to process sentiment analysis on a text, but Kimia Aksir recommended to me not to do that... [4].

Reflection on your work and establish your arguments (You can provide a good connection between what you proposed in Coursework 2 and how you have achieved those goals in Coursework 3.

Both my programs work well, especially the second. The patient in the second code can read the table and introduce the symptoms translated to number(s). Each symptom has a description in the initial dataset and could be included in the program/code. Obviously, a human doctor is much better and faster for this job.

The same should be done when the model output a number. It needs to be translated from number to word(s). This could be done automatically... And there is a file with data about each disease with its precautions to take, 4 in total. Recommendations from the doctor in a human-readable format could be implemented in future.

The last block of code is just to see 100% accuracy with any algorithm.

Generally, my work is well implemented. And I did NOT copy from Kaggle. There are 15 code solutions [5]. A little bit more organisation will be great to see in my whole code. And a nice UX design will be great as well to make the chatbot lovely to the patient.

References.

1.

https://scikit-learn.org/stable/modules/naive_bayes.html#naive-bayes

2.

<http://daydreamingnumbers.com/blog/introduction-naive-bayes-algorithm-python/>

3.

<https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn>

4.

<https://www.analyticsvidhya.com/blog/2021/07/performing-sentiment-analysis-with-naive-bayes-classifier/>

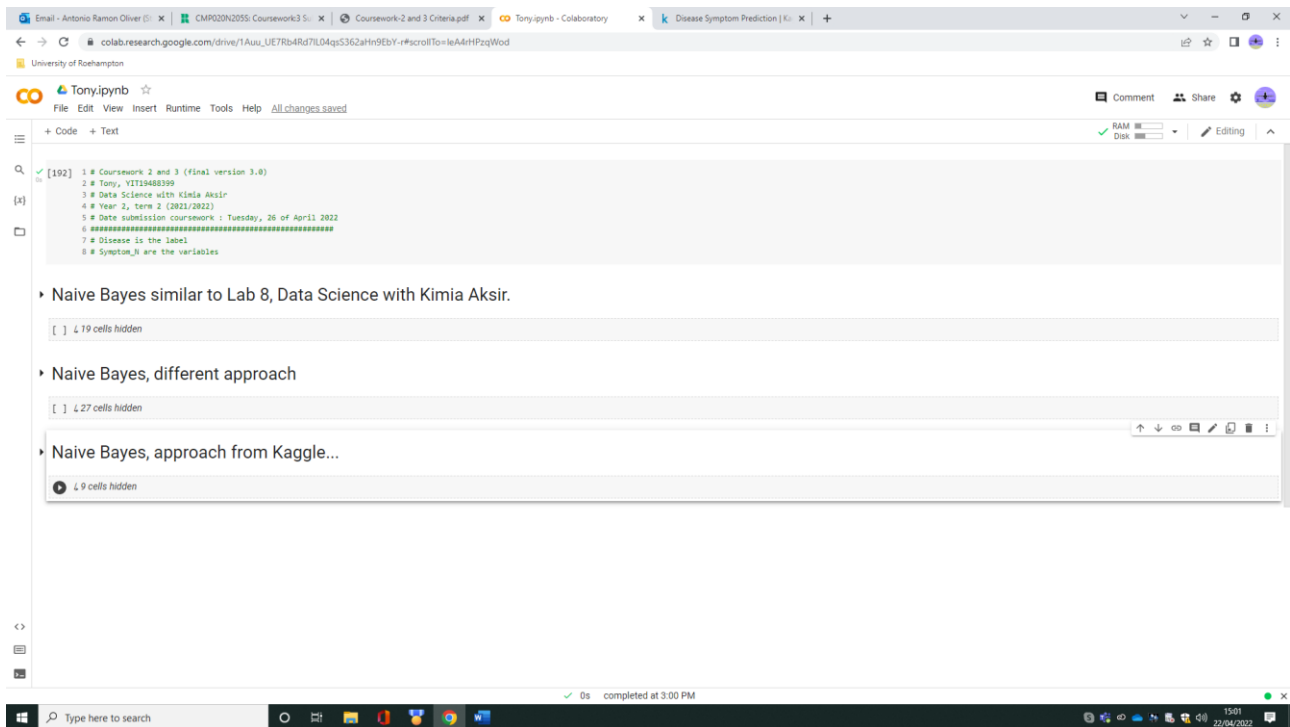
5.









<https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset/code?datasetId=672162&sortBy=voteCount>

6.

<https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset/code>

Screenshots and pictures.



Name	Status	Date modified	Type	Size
 dataset		29/03/2022 11:18	Microsoft Excel C...	618 KB
 symptom_Description		29/03/2022 11:18	Microsoft Excel C...	11 KB
 symptom_precaution		29/03/2022 11:18	Microsoft Excel C...	4 KB
 Symptom-severity		29/03/2022 11:18	Microsoft Excel C...	3 KB

This is all the files from Kaggle. “dataset” is the main used. “symptom_Description” and “symptom_precaution” are very useful for better implementation of the chatbot doctor Tony. The last dataset is not used, it is not mentioned in my work and can be discarded at this level/stage of my job.

University of Roehampton

Tony.pynb

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

RAM Disk

Editing

```

[287] 1 clf = CategoricalNB()
      2 clf.fit(X_train, y_train)

CategoricalNB()

1 print(X_train)
2 print(y_train)
3 print(len(X_train))
4 print(len(y_train))
5 y_pred = clf.predict(X_test)
6 print(y_pred)

[[ 7 7 28]
 [47 47 47]
 [28 11 12]
 ...
 [11 11 28]
 [15 15 28]
 [ 3 22 42]]
[ 6 20 35 ... 23 19 10]
1574
1574
[23 7 37 ... 6 21 17]

[289] 1 ac = accuracy_score(y_test, y_pred)
      2 print(ac)

0.4123729826658695

[210] 1 new_row = array(['itching', 'vomiting', 'yellowish_skin'])
      2
      3 new_row_converted = le.fit_transform(new_row)
      4 new_row_converted = new_row_converted.reshape(1,3)
      5 print(new_row_converted.shape)
      6 y_pred = clf.predict(new_row_converted)
      7 print(y_pred)
      8 print(y.take(y_pred))

(1, 3)
[30]
['Chronic cholestasis']

```

41%

0.4123729826658695

completed at 3:00 PM

University of Roehampton

Tony.pynb

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

RAM Disk

Editing

```

[287] 9 nbdf['Symptom_7'] = number.fit_transform(nbdf['Symptom_7'])
      10 nbdf['Symptom_8'] = number.fit_transform(nbdf['Symptom_8'])
      11 nbdf['Symptom_9'] = number.fit_transform(nbdf['Symptom_9'])
      12 nbdf['Symptom_10'] = number.fit_transform(nbdf['Symptom_10'])
      13 nbdf['Symptom_11'] = number.fit_transform(nbdf['Symptom_11'])
      14 nbdf['Symptom_12'] = number.fit_transform(nbdf['Symptom_12'])
      15 nbdf['Symptom_13'] = number.fit_transform(nbdf['Symptom_13'])
      16 nbdf['Symptom_14'] = number.fit_transform(nbdf['Symptom_14'])
      17 nbdf['Symptom_15'] = number.fit_transform(nbdf['Symptom_15'])
      18 nbdf['Symptom_16'] = number.fit_transform(nbdf['Symptom_16'])
      19 nbdf['Symptom_17'] = number.fit_transform(nbdf['Symptom_17'])

[288] 1 features = ['Symptom_1', 'Symptom_2', 'Symptom_3', 'Symptom_4', 'Symptom_5', 'Symptom_6', 'Symptom_7', 'Symptom_8',
      2           'Symptom_10', 'Symptom_11', 'Symptom_12', 'Symptom_13', 'Symptom_14', 'Symptom_15', 'Symptom_16', 'Symptom_17']
      3 target = 'Disease'

[289] 1 features_train, features_test, target_train, target_test = train_test_split(nbdf[features],
      2 nbdf[target], test_size = 0.70, random_state = 54)

[290] 1 model = GaussianNB()
      2 model.fit(features_train, target_train)

GaussianNB()

[291] 1 pred = model.predict(features_test)
      2 accuracy = accuracy_score(target_test, pred)
      3 print(accuracy)

0.8768873403819745

[335] 1 #Examples of symptoms input by user and first column is the disease number that can be translated to text...
      2 features_test

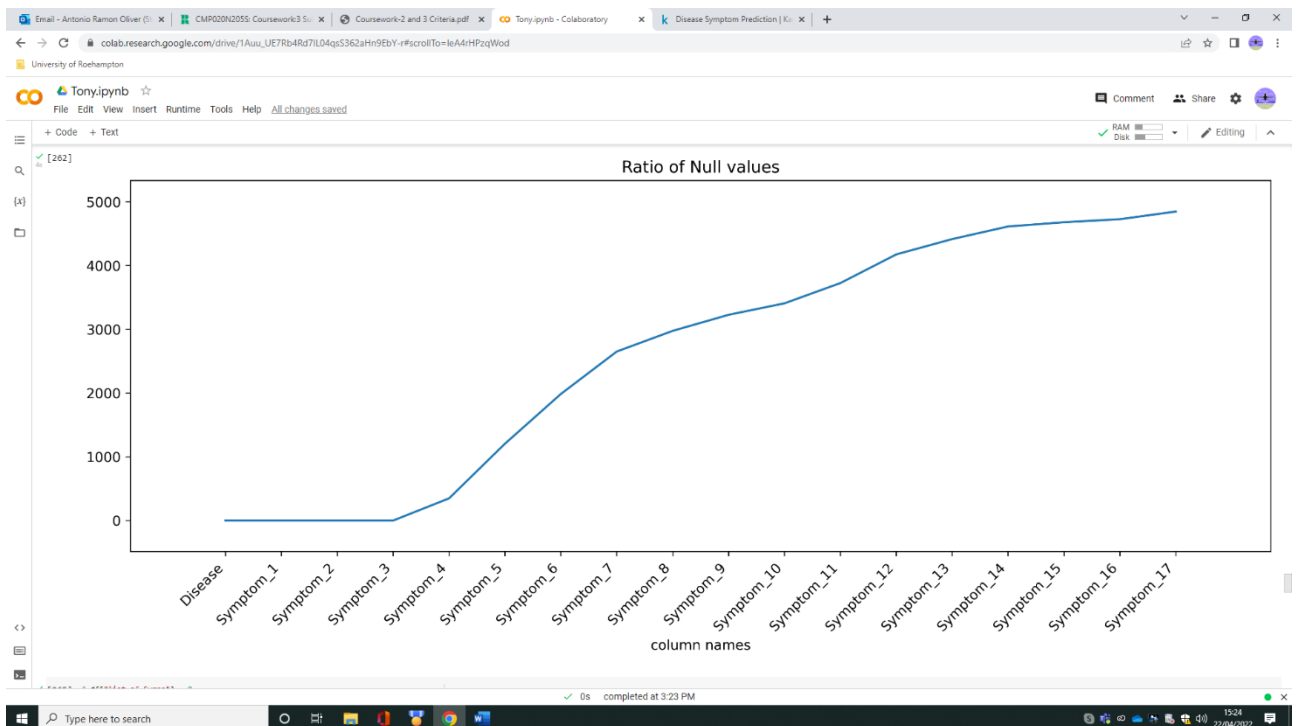
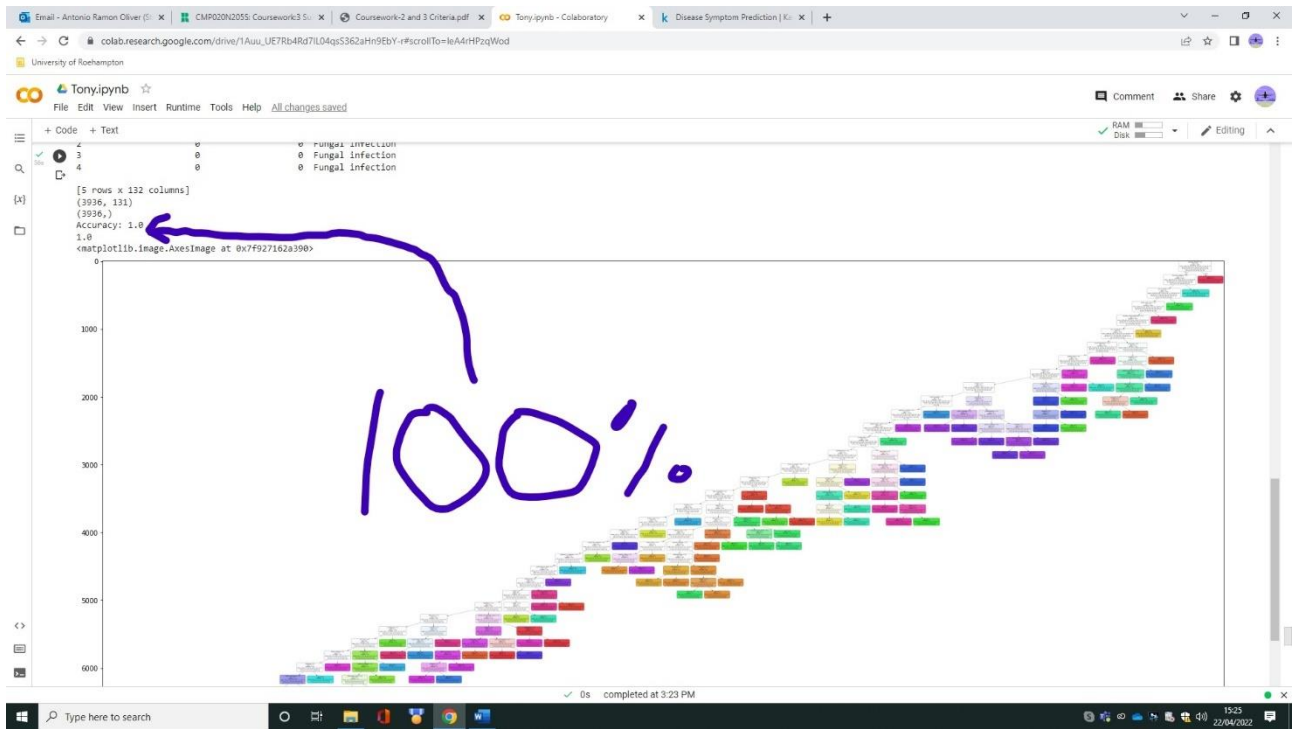
Symptom_1 Symptom_2 Symptom_3 Symptom_4 Symptom_5 Symptom_6 Symptom_7 Symptom_8 Symptom_9 Symptom_10 Symptom_11 Symptom_12 Symptom_13 Symptom_14 Symptom_15 Symptom_16 Symptom_17
1613      4      16      13      50      38      32      26      21      22      21      18      11      8      4      3      3      1
2713     11      46      39      24      17      1      18      6      6      11      18      11      8      4      3      3      1
2408     18      36      46      28      25      32      26      21      22      21      18      11      8      4      3      3      1
342      19      21      25      41      25      32      26      21      22      21      18      11      8      4      3      3      1

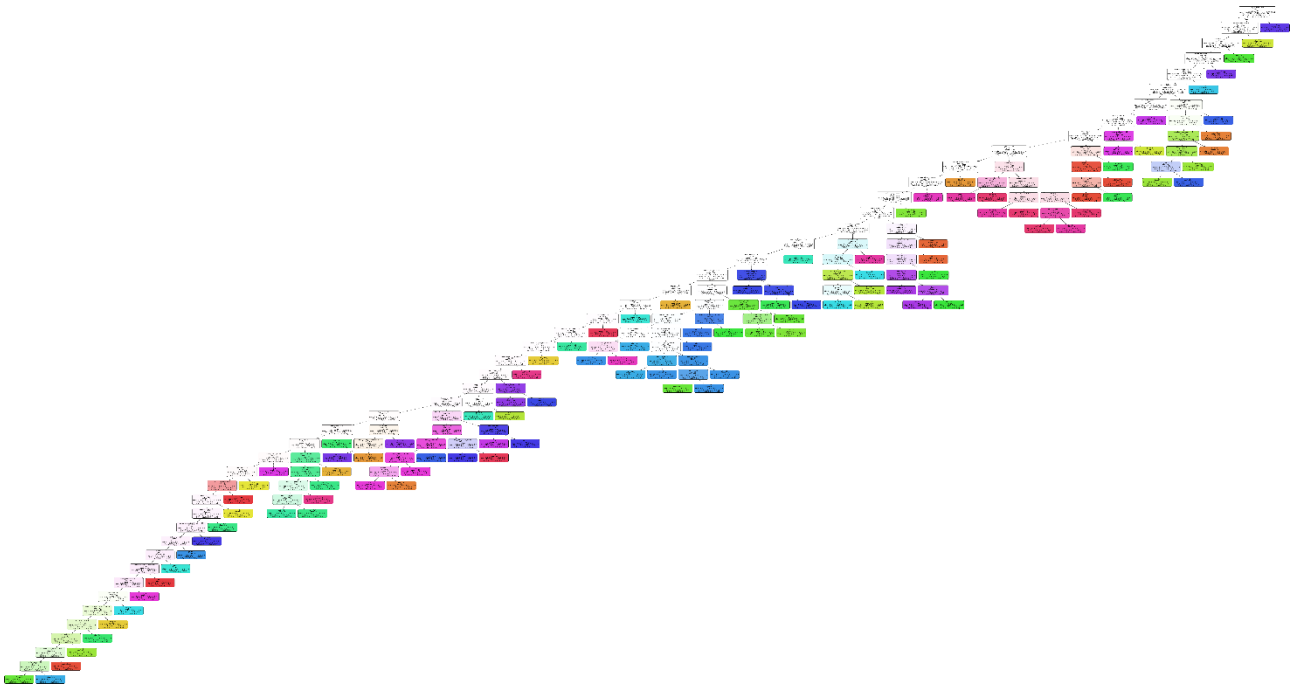
```

87%

0.8768873403819745

completed at 3:23 PM





#####END#####



Data Science

[This Photo](#) by Unknown Author is licensed under [CC BY-SA-NC](#)