National College of Ireland

Configuration Manual

MSc Internship
Cyber Security

Onkar Vilas Bhanarkar
Student ID: X19114761

School of Computing
National College of Ireland

Supervisor: Dr. Imran Khan

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Onkar Vilas Bhanarkar
Student ID:	X19114761
Programme:	Cyber Security
Year:	2020
Module:	MSc Internship
Supervisor:	Dr. Imran Khan
Submission Due Date:	17/08/2020
Project Title:	Configuration Manual
Word Count:	489
Page Count:	10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature:	
Date:	16th August 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Onkar Vilas Bhanarkar
X19114761

1 Introduction

The configuration manual provides the implementation phases and walkthrough of the research **"The mechanism to detect spam emails in Marathi language using NLP"**. We have used natural language processing libraries for the Marathi language such as INLTK and CLTK. also, machine learning algorithms like Naïve Bayes, Logistic regression, K-nearest neighbors, Decision tree classifier, Random Forest, Support vector machine, and Stochastic Gradient Descent. In the next sections, basic requirement and code walkthrough described briefly.

2 System Specification

The following software and Hardware requirement are needed to configure and implement this project.

2.1 Hardware Configuration

- Operating System: Windows 10 Pro
- Processor: Intel i5 8th Generation
- RAM: 8GB

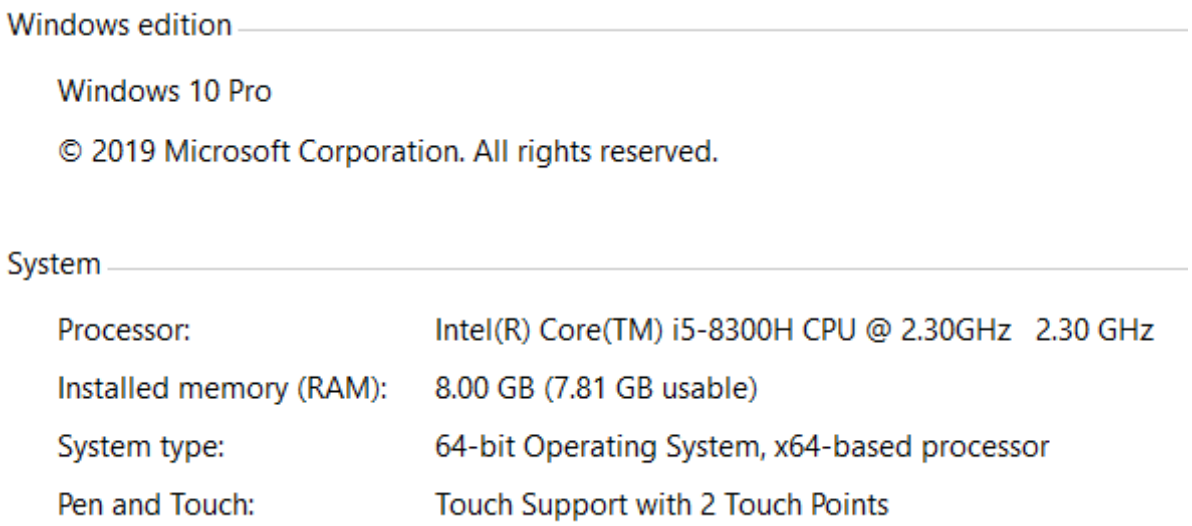

Figure 1: Hardware Specification

2.2 Software Configuration

Google Colab: we have used the Google-Colab platform, which provides pre-installed basic libraries which require in machine learning programs. It Is allowed to edit and run programs through the browser.

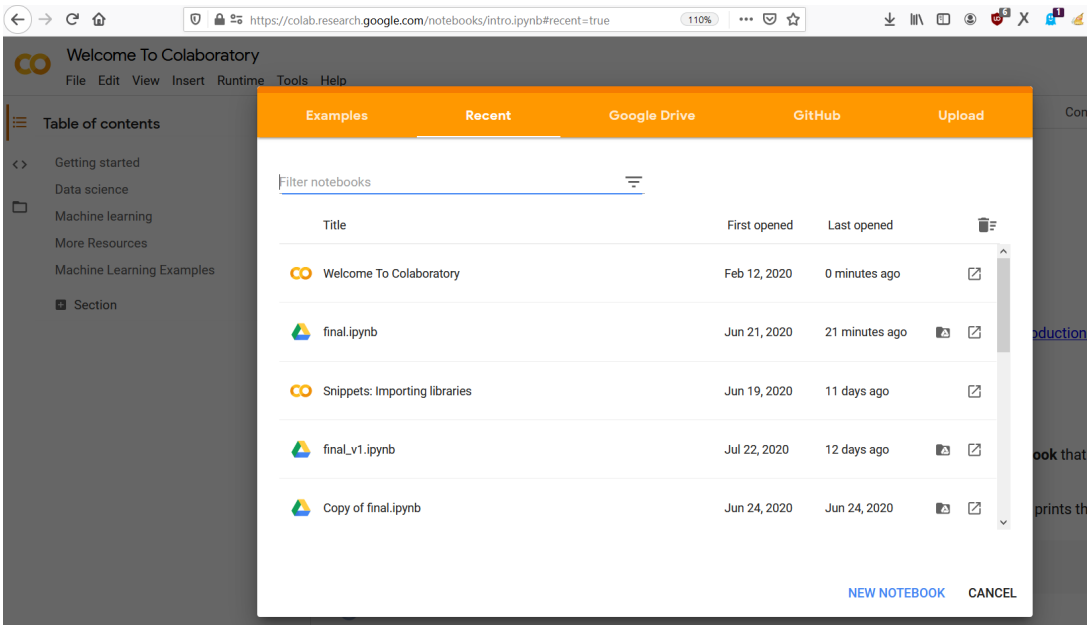

Figure 2: Google colab Notebook

3 Data Processing

3.1 Data Uploading and Preview

Our data in the form of .csv. The preview of data shown in figure.

```
[ ]  import pandas as pd
     import numpy as np

     df = pd.read_table('emails', header=None, encoding='utf-8')
```

Figure 3: Data Uploading

```
print(df.info())
print(df.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5569 entries, 0 to 5568
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       5569 non-null   object
 1   1       5569 non-null   object
dtypes: object(2)
memory usage: 87.1+ KB
None
      0                                                  1
0   ham  जुराँग बिंदू, वेडा पर्यंत जा .. फक्त Bugis n म...
1   ham            ठीक आहे Lar ... oni नाम wif गंमत करत ...
2  spam  2 wkly प्रगतिशील मध्ये मोफत नोंद नोंद प्रश्न प...
3   ham  यू करडा रंग म्हणून लवकर होर ... यू आधीच नंतर म...
4   ham  शं मला वाटत नाही तो usf ला, तो येथे सुमारे तरी...
```

Figure 4: Data Preview

3.2 Data Pre-processing

- Label Encoding : we have converted the ham and spam into 0 and 1 respectively.

```
[ ]

    from sklearn.preprocessing import LabelEncoder


    encoder = LabelEncoder()
    Y = encoder.fit_transform(classes)

    print(Y[:10])
```

```
[0 0 1 0 0 1 0 0 1 1]
```

Figure 5: Label Encoding

- Removing Unwanted information : Regular expressions are used to remove the unwanted information like Email address, web address, Currency symbol, Phone Number, English Words, and symbols from the data.

```
#Email Address
processed = text_messages.str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',
                                       '')
#Webaddresses
processed = processed.str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                   '')
#Currency Symbols
processed = processed.str.replace(r'£|\$', '')

#Phone Number (include paranthesis, spaces, no spaces, dashes)
processed = processed.str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                   '')
#Number
processed = processed.str.replace(r'\d+(\.\d+)?', '')

#English Words
processed = processed.str.replace(r'[a-zA-Z]','')

#Symbols
processed = processed.str.replace(r'[!@#$%^&*()_+,.;?]','')
print(processed)
```

```
0       जुराँग बिंदू वेडा पर्यंत जा  फक्त   महान जागति...
1                                 ठीक आहे    नाम  गंमत करत
2         प्रगतिशील मध्ये मोफत नोंद नोंद प्रश्न प्राप्...
3       यू करडा रंग म्हणून लवकर होर  यू आधीच नंतर म्हण...
4       शं मला वाटत नाही तो  ला तो येथे सुमारे तरी आयुष्य
                               ...
```

Figure 6: Removing Unwanted Information

- Stop Words: We have used the CLTK library for removing the stopwords[1].

```
from cltk.stop.marathi.stops import STOP_LIST
#print(STOP_LIST[1])

processed = processed.apply(lambda x: ' '.join(
    term for term in x.split() if term not in STOP_LIST[:93]))
```

Figure 7: Stopwords Removal

3.3 Tokenization and Feature Extraction

In this section we have used the INLTK Library[2] for tokenization.

```
[ ]  from inltk.inltk import tokenize

     all_words = []

     for message in processed:

         words=tokenize(message ,'mr')
         for w in words:
             all_words.append(w)

     all_words = nltk.FreqDist(all_words)
```

Figure 8: Tokenization

- Feature selection : In all, we found total 5686 common words. Among them, we have selected the top 1500 common words as a feature.

```
print('Number of words: {}'.format(len(all_words)))
print('Most common words: {}'.format(all_words.most_common(100)))
```

```
Number of words: 5686
Most common words: [('_एक', 1458), ('_नाही', 1241), ('_', 1226), ('_मला', 954), ('_नाम', 666), ('_करू', 609), ('_कॉल', 591),
```

Figure 9: Common Words

The following code is used to extract the feature.

```
[ ]  word_features = list(all_words.keys())[:1500]
```

```
[ ]  def find_features(message):
         words=tokenize(message ,'mr')
         features = {}
         for word in word_features:
             features[word] = (word in words)

         return features
     features = find_features(processed[0])
     for key, value in features.items():
         if value == True:
             print (key)
```

```
_जु
र
ाँग
_बिंदू
_वेडा
_पर्यंत
_जा
_फक्त
_महान
_जागतिक
_
```

Figure 10: Feature Extraction

4 Machine learning Algorithms

We have used multiple machine learning algorithms to evaluate the accuracy. before that we have divided our pre-processed data into 80:20, training and testing respectively.

```
[ ] from sklearn import model_selection


    training, testing = model_selection.train_test_split(featuresets, test_size = 0.25, random_state=seed)
```

```
[ ]

    print(len(training))
    print(len(testing))
```

```
4176
1393
```

Figure 11: Training and Testing Dataset

For the machine learning algorithms we have used the following python libraries.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
```

Figure 12: Python Libraries

The below figure shows the implementation of machine learning algorithms respectively.

```
names = ["K Nearest Neighbors", "Decision Tree", "Random Forest", "Logistic Regression", "SGD Classifier",
         "Naive Bayes", "SVM Linear"]

classifiers = [
    KNeighborsClassifier(),
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    LogisticRegression(),
    SGDClassifier(max_iter = 100),
    MultinomialNB(),
    SVC(kernel = 'linear')
]

models = zip(names, classifiers)
txt_features, labels = zip(*testing)
for name, model in models:
    nltk_model = SklearnClassifier(model)
    nltk_model.train(training)
   # txt_features, labels = zip(*testing)
    accuracy = nltk.classify.accuracy(nltk_model, testing)*100
    print("{} Accuracy: {}".format(name, accuracy))
    prediction = nltk_model.classify_many(txt_features)
    print(classification_report(labels, prediction))
    print(pd.DataFrame(
      confusion_matrix(labels, prediction),
      index = [['actual', 'actual'], ['ham', 'spam']],
      columns = [['predicted', 'predicted'], ['ham', 'spam']])   )
    print("")
    print("")
```

Figure 13: Model Training

5 Evaluation

We have used sklearn.metrics for the evaluation. The following figure shows the accuracy, precision, recall, f1 score respectively.

```
Logistic Regression Accuracy: 98.27709978463747
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1199
           1       0.96      0.92      0.94       194

    accuracy                           0.98      1393
   macro avg       0.97      0.96      0.96      1393
weighted avg       0.98      0.98      0.98      1393

            predicted
                  ham spam
actual ham       1191    8
       spam        16  178
```

Figure 14: Evaluation

6 References

[1]"The Classical Language Toolkit", Cltk.org, 2020. [Online]. Available: http://cltk.org/.

[2]"Natural Language Toolkit for Indic Languages iNLTK latest documentation", Inltk.readthedocs.io, 2020. [Online]. Available: https://inltk.readthedocs.io/en/latest/.