RAMAKRISHNA MISSION VIVEKANANDA EDUCATIONAL AND RESEARCH INSTITUTE

(Accredited by NAAC with 'A++' Grade)
Coimbatore – 641 020

in Collaboration with

Cognizant

SCHOOL OF MATHEMATICAL SCIENCE

SEPTEMBER – 2020

DEPARTMENT OF COMPUTER SCIENCE
CENTRE OF DATA SCIENCE

NAME	**: Swapnil Pradeep**
REG. NO	**: h19msds012**
PROGRAMME	**: M.Sc. (Data Science)**
SEMESTER	**: 3st Semester**
PROJECT TITLE	**: HOTEL REVIEW**
COURSE CODE	**: 19DS2P06**

RAMAKRISHNA MISSION VIVEKANANDA EDUCATIONAL AND RESEARCH INSTITUTE

(Accredited by NAAC with 'A++' Grade)
Coimbatore – 641 020

in Collaboration with

Cognizant

SCHOOL OF MATHEMATICAL SCIENCE

SEPTEMBER – 2020

DEPARTMENT OF COMPUTER SCIENCE
CENTRE OF DATA SCIENCE

Bonafide Certificate

This is to certify that the project work done by **Swapnil Pradeep** (**H19MSDS012**) entitled "**HOTEL REVIEW**" in his third Semester during the academic year 2020-2021. Submitted for the Semester viva-voice Examination held on 22-01-2021.

Staff In-Charge

Head of the Department

Internal Examiner

External Examiner

DECLARATION

I hereby declare that this project entitled " HOTEL REVIEW " submitted to Ramakrishna Mission Vivekananda Educational and Research Institute, Coimbatore-20. is partial fulfillment of the requirement of the degree in M.Sc., (Data Science) is a record of the original project done by me under the guidance of Dr.R.Sridhar M.Sc., MCA., M.Phil., Ph.D., Professor & Head, Ramakrishna Mission Vivekananda Educational And Research Institute, Coimbatore - 641 020.

Place : Coimbatore
Date :

Signature of the Candidate

STUDENT NAME

(H19MSDS000)

ACKNOWLEDGEMENT

My pranams to **Rev. Swami Garishthananda**, Secretary, Ramakrishna Mission Vidyalaya and **Rev. Swami Anapekshanada**, Asst. Administrative Head, FyCSAR, Ramakrishna Mission Vivekananda Educational and Research Institute, Coimbatore - 20 for providing me facilities and encouragement to complete the project work.

I express my profound gratitude to **Dr. R.Sridhar,** Professor & Head, FyCSAR, RKMVERI, Coimbatore-20 for his whole hearted encouragement and timely help.

I placed on record my heartfelt thanks and gratitude to **Sri. D. Raja,** Project Consultant and **Sri. V.Dineshkumar,** Assistant Professor, FyCSAR, RKMVERI, Coimbatore-20 under whose guidance the work has been done. It is a matter of joy that without his valuable suggestions, able guidance, scholarly touch and piercing insight that he offered me in each and every stage of this study, coupled with his unreserved, sympathetic and encouraging attitude, this thesis could not have been presented in this manner.

I am thankful to the following members from the industry for their multi-dimensional review and support to complete my project work.

1. Rohini Krishnan, Senior Manager, Cognizant
2. Manjunaath Alagiriswamy, Senior Associate, Cognizant
3. Muralidharan Sivakumar, Senior Associate, Cognizant
4. Kulshum Azmi, Behavioural Trainer, Cognizant
5. Rekha Priyadharshini, Python SME, Cognizant
6. Rajasekar, Pyhton SME, Cognizant
7. Hemnath Muthukrushnun, Data Scientist SME, Cognizant
8. Shabarivasan RC, Data Scientist SME, Cognizant
9. Balasubramanian Mahadevan, Managing Partner, Algolitics India LLP, Coimbatore

SYNOPSIS

This is a NLP project which consist of some hotel reviews data . Each observation consists in one customer review for one hotel. Each customer review is composed of a textual feedback of the customer's experience at the hotel and an overall rating. For each textual review, I want to predict if it corresponds to a good review (the customer is happy) or to a bad one (the customer is not satisfied) . This will be done by Sentiment analysis is part of the Natural Language Processing (NLP) techniques that consists in extracting emotions related to some raw texts. This is usually used on social media posts and customer reviews in order to automatically understand if some users are positive or negative and why.

HOTEL REVIEW PROJECT

STRATEGIC GOAL

To know the hotel review is it good or bad

OBJECTIVE

Recommend hotel to improve facilities

importing packages

In [19]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import spacy
import nltk
```

In [3]:

```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\swapy\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[3]:

```
True
```

In [4]:

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\swapy\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[4]:

```
True
```

In [ ]:

```

```

load data

In [5]:

```
df = pd.read_excel('Downloads\\DEMODATASET.xlsx')
```

In [6]:

```
df.head()
```

Out[6]:

	hotelcode	review_content	reviewid	id	sentiment_score	strength	polarity	p
0	332424	Good stay	151518420190623	1515184	NaN	NaN	NaN	
1	1887752	Place is great. Location is fabulous. It's rig...	151518520201201	1515185	NaN	NaN	NaN	
2	1259085	Very nice place, I have visited the Jim Corbet...	151539220171007	1515392	NaN	NaN	NaN	
3	2384	I sent 4 times e-mail and there is no answer :(	151539320150410	1515393	NaN	NaN	NaN	
4	1248837	Good rooms and service, but located in a busy ...	151629820190623	1516298	NaN	NaN	NaN	

In [7]:

```
df.shape
```

Out[7]:

```
(6348, 8)
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
hotelcode             0
review_content        0
reviewid              0
id                    0
sentiment_score    6348
strength           6348
polarity           6348
properties         6348
dtype: int64
```

In [9]:

```
df.loc[df["review_content"] == ""] #Checks for empty review strings
```

Out[9]:

	hotelcode	review_content	reviewid	id	sentiment_score	strength	polarity	properties

cleaning data set

In [10]:

```
# remove 'No Negative' or 'No Positive' from text
df["review_content"] = df["review_content"].apply(lambda x: x.replace("No Negative", "").re
```

If the user doesn't leave any negative feedback comment, this will appear as "No Negative" in our data. This is the same for the positive comments with the default value "No Positive". We have to remove those parts from our texts.

In [11]:

```
from nltk.corpus import wordnet

def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN
```

In [12]:

```
import string
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer
```

In [13]:

```
def clean_text(text):
    # lower text
    text = text.lower()
    # tokenize text and remove puncutation
    text = [word.strip(string.punctuation) for word in text.split(" ")]
    # remove words that contain numbers
    text = [word for word in text if not any(c.isdigit() for c in word)]
    # remove stop words
    stop = stopwords.words('english')
    text = [x for x in text if x not in stop]
    # remove empty tokens
    text = [t for t in text if len(t) > 0]
    # pos tag text
    pos_tags = pos_tag(text)
    # lemmatize text
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]
    # remove words with only one letter
    text = [t for t in text if len(t) > 1]
    # join all
    text = " ".join(text)
    return(text)
```

To clean textual data, here i used theese all functions

lower the text , tokenize the text (split the text into words) and remove the punctuation, remove useless words that contain numbers, remove useless stop words like 'the', 'a' ,'this' etc, Part-Of-Speech (POS) tagging: assign a tag to every word to define if it corresponds to a noun, a verb etc. using the WordNet lexical database, lemmatize the text: transform every word into their root form (e.g. rooms -> room, slept -> sleep).

In [16]:

```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\swapy\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\wordnet.zip.
```

Out[16]:

```
True
```

In [17]:

```
df["clean"] = df["review_content"].apply(lambda x: clean_text(x))
```

In [18]:

```
df.head()
```

Out[18]:

	hotelcode	review_content	reviewid	id	sentiment_score	strength	polarity	p
0	332424	Good stay	151518420190623	1515184	NaN	NaN	NaN	
1	1887752	Place is great. Location is fabulous. It's rig...	151518520201201	1515185	NaN	NaN	NaN	
2	1259085	Very nice place, I have visited the Jim Corbet...	151539220171007	1515392	NaN	NaN	NaN	
3	2384	I sent 4 times e-mail and there is no answer :(	151539320150410	1515393	NaN	NaN	NaN	
4	1248837	Good rooms and service, but located in a busy ...	151629820190623	1516298	NaN	NaN	NaN	

Sentiment analysis

In [21]:

```
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\swapy\AppData\Roaming\nltk_data...
```

Out[21]:

```
True
```

In [22]:

```
from nltk.sentiment.vader import SentimentIntensityAnalyzer

sid = SentimentIntensityAnalyzer()
df["sentiments"] = df["review_content"].apply(lambda x: sid.polarity_scores(x))
df = pd.concat([df.drop(['sentiments'], axis=1), df['sentiments'].apply(pd.Series)], axis=1
```

Here i take sentiment analysis features because we can guess that customers reviews are highly linked to how they felt about their stay at the hotel. Here We use Vader, which is a part of the NLTK module designed for sentiment analysis. Vader uses a lexicon of words to find which ones are positives or negatives. It also takes into accout the context of the sentences to determine the sentiment scores.For each text, Vader retuns 4 values. A neutrality score, a positivity score, a negativity score,an overall score that summarizes the previous scores,We will integrate those 4 values as features in our dataset.

In [23]:

```
# add number of characters column
df["nb_chars"] = df["review_content"].apply(lambda x: len(x))

# add number of words column
df["nb_words"] = df["review_content"].apply(lambda x: len(x.split(" ")))
```

In [26]:

```
pip install gensim
```

```
Collecting gensim
  Downloading gensim-3.8.3-cp37-cp37m-win_amd64.whl (24.2 MB)
Requirement already satisfied: numpy>=1.11.3 in c:\users\swapy\anaconda3\lib
\site-packages (from gensim) (1.18.1)
Requirement already satisfied: scipy>=0.18.1 in c:\users\swapy\anaconda3\lib
\site-packages (from gensim) (1.4.1)
Collecting Cython==0.29.14
  Downloading Cython-0.29.14-cp37-cp37m-win_amd64.whl (1.7 MB)
Requirement already satisfied: six>=1.5.0 in c:\users\swapy\anaconda3\lib\si
te-packages (from gensim) (1.14.0)
Collecting smart-open>=1.8.1
  Downloading smart_open-4.1.0.tar.gz (116 kB)
Building wheels for collected packages: smart-open
  Building wheel for smart-open (setup.py): started
  Building wheel for smart-open (setup.py): finished with status 'done'
  Created wheel for smart-open: filename=smart_open-4.1.0-py3-none-any.whl s
ize=106210 sha256=85e48f2e1b3beb40eb4f606293fb2b874280b2f44ca32088d50395b3a1
f92af5
  Stored in directory: c:\users\swapy\appdata\local\pip\cache\wheels\12\af\e
0\2adc6007eb052d6eb4ac8f01c5f699e4bc479f5ba764aadeb6
Successfully built smart-open
Installing collected packages: Cython, smart-open, gensim
  Attempting uninstall: Cython
    Found existing installation: Cython 0.29.15
    Uninstalling Cython-0.29.15:
      Successfully uninstalled Cython-0.29.15
Successfully installed Cython-0.29.14 gensim-3.8.3 smart-open-4.1.0
Note: you may need to restart the kernel to use updated packages.
```

In [27]:

```
# create doc2vec vector columns
from gensim.test.utils import common_texts
from gensim.models.doc2vec import Doc2Vec, TaggedDocument

documents = [TaggedDocument(doc, [i]) for i, doc in enumerate(df["clean"].apply(lambda x: x

# train a Doc2Vec model with our text data
model = Doc2Vec(documents, vector_size=5, window=2, min_count=1, workers=4)

# transform each document into a vector data
doc2vec_df = df["clean"].apply(lambda x: model.infer_vector(x.split(" "))).apply(pd.Series)
doc2vec_df.columns = ["doc2vec_vector_" + str(x) for x in doc2vec_df.columns]
reviews_df = pd.concat([df, doc2vec_df], axis=1)
```

the next step consist in extracting vector representations for every review. The module Gensim creates a numerical vector representation of every word in the corpus by using the contexts in which they appear (Word2Vec). This is performed using shallow neural networks. What's interesting is that similar words will have similar representation vectors.

Each text can also be transformed into numerical vectors using the word vectors (Doc2Vec). Same texts will also have similar representations and that is why we can use those vectors as training features.

We first have to train a Doc2Vec model by feeding in our text data. By applying this model on our reviews, we can get those representation vectors.

In [28]:

```
# add tf-idfs columns
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(min_df = 10)
tfidf_result = tfidf.fit_transform(df["clean"]).toarray()
tfidf_df = pd.DataFrame(tfidf_result, columns = tfidf.get_feature_names())
tfidf_df.columns = ["word_" + str(x) for x in tfidf_df.columns]
tfidf_df.index = df.index
df = pd.concat([df, tfidf_df], axis=1)
```

In [34]:

```
df.head()
```

Out[34]:

	hotelcode	review_content	reviewid	id	sentiment_score	strength	polarity	p
0	332424	Good stay	151518420190623	1515184	NaN	NaN	NaN	
1	1887752	Place is great. Location is fabulous. It's rig...	151518520201201	1515185	NaN	NaN	NaN	
2	1259085	Very nice place, I have visited the Jim Corbet...	151539220171007	1515392	NaN	NaN	NaN	
3	2384	I sent 4 times e-mail and there is no answer :(	151539320150410	1515393	NaN	NaN	NaN	
4	1248837	Good rooms and service, but located in a busy ...	151629820190623	1516298	NaN	NaN	NaN	

5 rows × 2063 columns

In [48]:

```
df.columns
```

Out[48]:

```
Index(['hotelcode', 'review_content', 'reviewid', 'id', 'sentiment_score',
       'strength', 'polarity', 'properties', 'clean', 'neg',
       ...
       'word_zion', 'word_zone', 'word_zu', 'word_zum', 'word_за',
       'word_место', 'word_много', 'word_на', 'word_не', 'word_это'],
      dtype='object', length=2063)
```

In [35]:

```
df.to_excel(r'C:\Users\swapy\Downloads\\AAD.xlsx', index = False)
```

In [36]:

```
df.shape
```

Out[36]:

```
(6348, 2063)
```

Exploratory data analysis

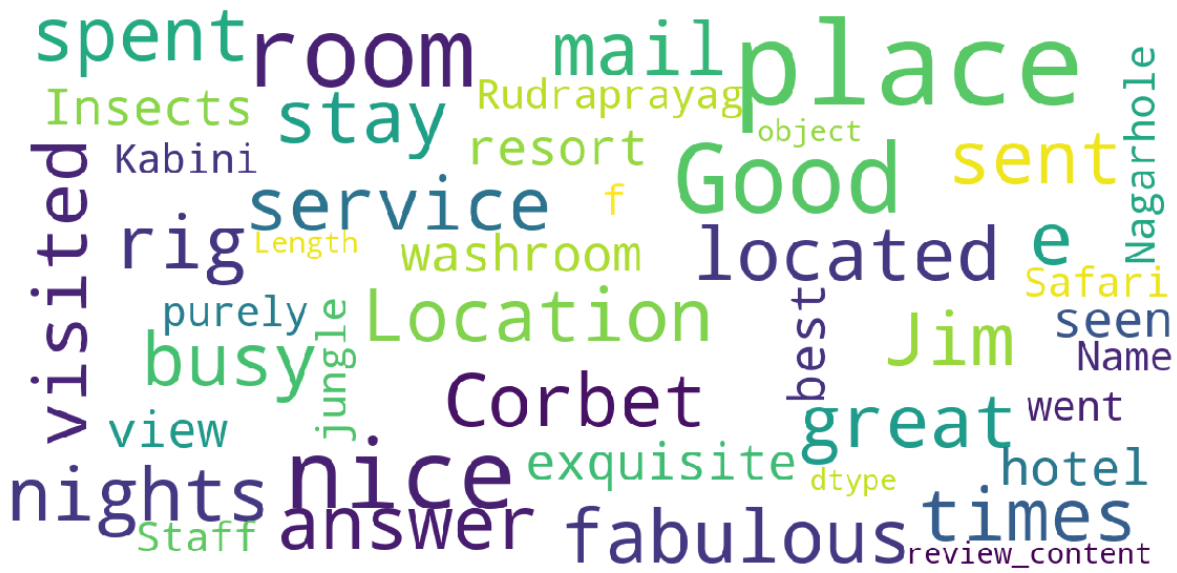
In [40]:

```
pip install wordcloud
```

```
Requirement already satisfied: wordcloud in c:\users\swapy\anaconda3\lib\sit
e-packages (1.8.1)
Requirement already satisfied: pillow in c:\users\swapy\anaconda3\lib\site-p
ackages (from wordcloud) (7.0.0)
Requirement already satisfied: numpy>=1.6.1 in c:\users\swapy\anaconda3\lib
\site-packages (from wordcloud) (1.18.1)
Requirement already satisfied: matplotlib in c:\users\swapy\anaconda3\lib\si
te-packages (from wordcloud) (3.1.3)
Requirement already satisfied: cycler>=0.10 in c:\users\swapy\anaconda3\lib
\site-packages (from matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\swapy\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (1.1.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\users\swapy\anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.
4.6)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\swapy\anacon
da3\lib\site-packages (from matplotlib->wordcloud) (2.8.1)
Requirement already satisfied: six in c:\users\swapy\anaconda3\lib\site-pack
ages (from cycler>=0.10->matplotlib->wordcloud) (1.14.0)
Requirement already satisfied: setuptools in c:\users\swapy\anaconda3\lib\si
te-packages (from kiwisolver>=1.0.1->matplotlib->wordcloud) (45.2.0.post2020
0210)
Note: you may need to restart the kernel to use updated packages.
```

In [41]:

```
# wordcloud function

from wordcloud import WordCloud
import matplotlib.pyplot as plt

def show_wordcloud(data, title = None):
    wordcloud = WordCloud(
        background_color = 'white',
        max_words = 200,
        max_font_size = 40,
        scale = 3,
        random_state = 42
    ).generate(str(data))

    fig = plt.figure(1, figsize = (20, 20))
    plt.axis('off')
    if title:
        fig.suptitle(title, fontsize = 20)
        fig.subplots_adjust(top = 2.3)

    plt.imshow(wordcloud)
    plt.show()

# print wordcloud
show_wordcloud(df["review_content"])
```



Most of the words are indeed related to the hotels: room, staff, breakfast, etc. Some words are more related to the customer experience with the hotel stay: perfect, loved, expensive, dislike, etc.

In [43]:

```
# highest positive sentiment reviews (with more than 5 words)
df[df["nb_words"] >= 5].sort_values("pos", ascending = False)[["review_content", "pos"]].he
```

Out[43]:

	review_content	pos
770	Perfect settlement, nice n comfortable	0.907
3584	A paradise! Beautiful island !	0.897
3988	Very peaceful, beautiful n nice experience	0.843
3982	Superb hotel! All Good! :)	0.841
114	Spent great time fantastic holiday	0.839
3103	Awesome. Spend very nice holidays	0.834
3624	Good was great, nice room,	0.831
2526	Natural beauty ,very good service	0.826
1948	Awesome place. great food. Good n helpful staff	0.822
6271	Great natural fresh air chamber	0.817

The most positive reviews indeed correspond to some good feedbacks.

In [44]:

```
# lowest negative sentiment reviews (with more than 5 words)
df[df["nb_words"] >= 5].sort_values("neg", ascending = False)[["review_content", "neg"]].he
```

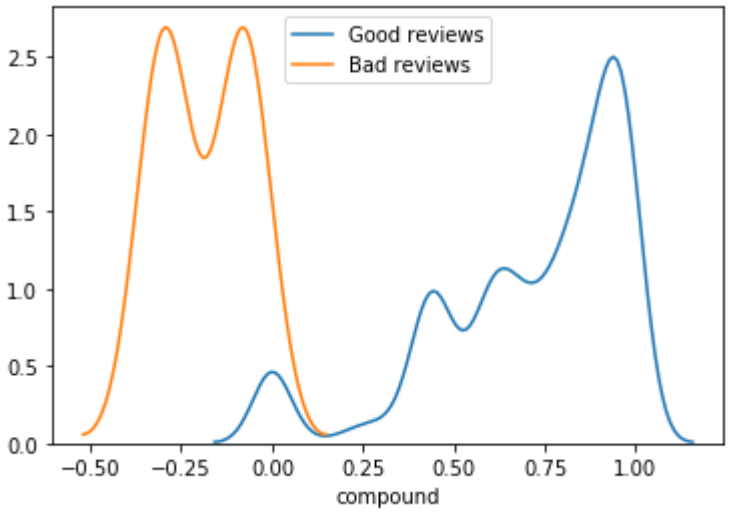
Out[44]:

	review_content	neg
4245	Pathetic rooms, poor air conditioning, useless...	0.706
631	very bad at this cost. service horrible.	0.593
2459	worst experience, not worth this triff	0.590
5296	No value of customers !!!! No service No resp...	0.589
6201	Taken 600 but hotel is worst. No TV, No Hot Wa...	0.539
612	Very bad service and charged extra money for n...	0.527
1042	worst service ever... no service no break fast...	0.517
6299	Very poor service, bad front desk, least flexi...	0.510
368	Worst place i had ever seen	0.506
6284	Very bad Beaver service capent	0.487

the most negative reviews.

In [75]:

```
# plot sentiment distribution for positive and negative reviews

import seaborn as sns

for x in [0, 1]:
    subset = df[df['neg'] == x]

    # Draw the density plot
    if x == 0:
        label = "Good reviews"
    else:
        label = "Bad reviews"
    sns.distplot(subset['compound'], hist = False, label = label)
```



conclusion

It is completely possible to use only raw text as input for making predictions. The most important thing is to be able to extract the relevant features from this raw source of data. This kind of data can often come as a good complementary source in data science projects in order to extract more learning features and increase the predictive power of the models.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: