EDS Activity 1

Name: *Dewesh Jagdish Barapatre*

Roll No.: *CS7-77*

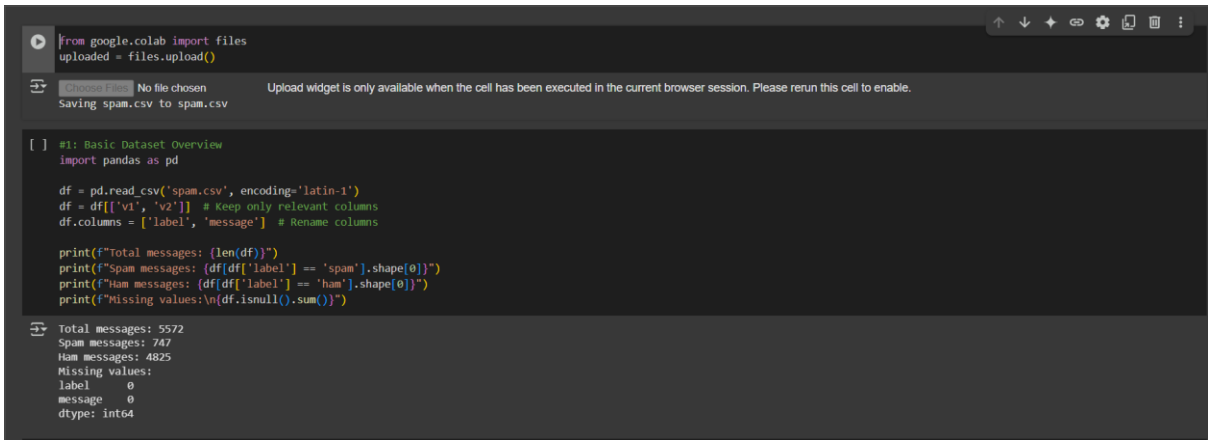
PRN: *202401110053*

Division: *CS7*

DataSet: *SMS Spam Collection*

Google Colab Link:

https://colab.research.google.com/drive/1nilaiagug-oFyikk0-1dmcIpXtEOYwjT?usp=sharing



```
from google.colab import files
uploaded = files.upload()
```

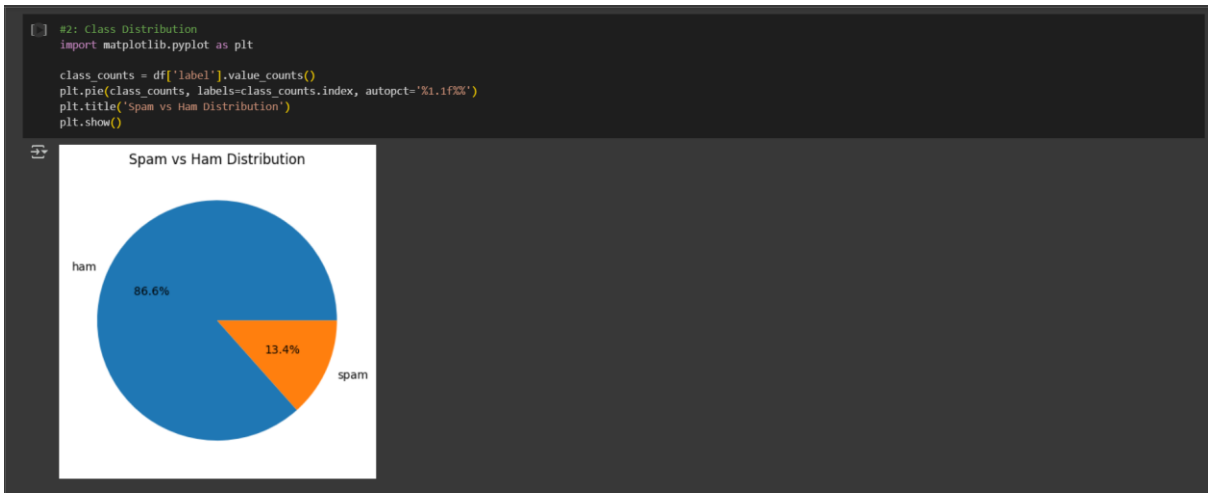
```
Choose Files  No file chosen
Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving spam.csv to spam.csv
```

```
#1: Basic Dataset Overview
import pandas as pd

df = pd.read_csv('spam.csv', encoding='latin-1')
df = df[['v1', 'v2']]  # Keep only relevant columns
df.columns = ['label', 'message']  # Rename columns

print(f"Total messages: {len(df)}")
print(f"Spam messages: {df[df['label'] == 'spam'].shape[0]}")
print(f"Ham messages: {df[df['label'] == 'ham'].shape[0]}")
print(f"Missing values:\n{df.isnull().sum()}")
```

```
Total messages: 5572
Spam messages: 747
Ham messages: 4825
Missing values:
label      0
message    0
dtype: int64
```



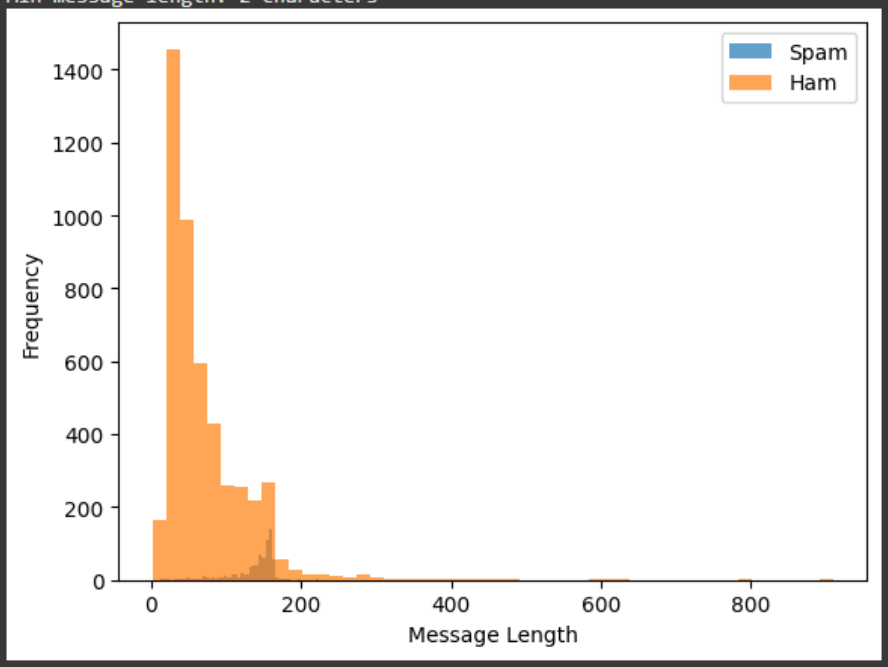
```
#2: Class Distribution
import matplotlib.pyplot as plt

class_counts = df['label'].value_counts()
plt.pie(class_counts, labels=class_counts.index, autopct='%1.1f%%')
plt.title('Spam vs Ham Distribution')
plt.show()
```

```python
#3: Message Length Analysis
df['length'] = df['message'].apply(len)
print(f"Average message length: {df['length'].mean():.2f} characters")
print(f"Max message length: {df['length'].max()} characters")
print(f"Min message length: {df['length'].min()} characters")

df[df['label'] == 'spam']['length'].plot(kind='hist', bins=50, alpha=0.7, label='Spam')
df[df['label'] == 'ham']['length'].plot(kind='hist', bins=50, alpha=0.7, label='Ham')
plt.legend()
plt.xlabel('Message Length')
plt.show()
```

```
Average message length: 80.12 characters
Max message length: 910 characters
Min message length: 2 characters
```



```
[ ] #4: Most Common Words in Spam
    from collections import Counter
    import re

    spam_words = ' '.join(df[df['label'] == 'spam']['message']).lower()
    words = re.findall(r'\w+', spam_words)
    common_spam = Counter(words).most_common(10)

    print("Top 10 words in spam messages:")
    for word, count in common_spam:
        print(f"{word}: {count}")
```

```
Top 10 words in spam messages:
to: 688
a: 377
call: 355
å: 299
you: 297
your: 264
free: 224
2: 206
the: 206
for: 203
```

```
[ ] #5: Most Common Words in Ham
    ham_words = ' '.join(df[df['label'] == 'ham']['message']).lower()
    words = re.findall(r'\w+', ham_words)
    common_ham = Counter(words).most_common(10)

    print("Top 10 words in ham messages:")
    for word, count in common_ham:
        print(f"{word}: {count}")
```

```
Top 10 words in ham messages:
i: 2940
you: 1943
to: 1554
the: 1122
a: 1056
u: 1018
and: 857
in: 818
me: 772
my: 750
```

```
[ ] #6: Presence of URLs in Messages
    url_pattern = r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+'
    df['has_url'] = df['message'].str.contains(url_pattern, regex=True)

    print(f"Spam messages with URLs: {df[(df['label'] == 'spam') & df['has_url']].shape[0]}")
    print(f"Ham messages with URLs: {df[(df['label'] == 'ham') & df['has_url']].shape[0]}")
```

```
Spam messages with URLs: 19
Ham messages with URLs: 0
```

```
[ ] #7: Presence of Phone Numbers
    phone_pattern = r'\b\d{10,13}\b'
    df['has_phone'] = df['message'].str.contains(phone_pattern, regex=True)

    print(f"Spam messages with phone numbers: {df[(df['label'] == 'spam') & df['has_phone']].shape[0]}")
    print(f"Ham messages with phone numbers: {df[(df['label'] == 'ham') & df['has_phone']].shape[0]}")
```

```
Spam messages with phone numbers: 377
Ham messages with phone numbers: 1
```

```
[ ] #8: Presence of Currency Symbols
    currency_pattern = r'£|\$|€'
    df['has_currency'] = df['message'].str.contains(currency_pattern, regex=True)

    print(f"Spam messages with currency symbols: {df[(df['label'] == 'spam') & df['has_currency']].shape[0]}")
    print(f"Ham messages with currency symbols: {df[(df['label'] == 'ham') & df['has_currency']].shape[0]}")
```

```
Spam messages with currency symbols: 257
Ham messages with currency symbols: 18
```

```
[ ] #9: Presence of Uppercase Words
    df['uppercase_ratio'] = df['message'].apply(lambda x: sum(1 for c in x if c.isupper()) / len(x) if len(x) > 0 else 0)

    print(f"Average uppercase ratio in spam: {df[df['label'] == 'spam']['uppercase_ratio'].mean():.4f}")
    print(f"Average uppercase ratio in ham: {df[df['label'] == 'ham']['uppercase_ratio'].mean():.4f}")
```

```
Average uppercase ratio in spam: 0.1106
Average uppercase ratio in ham: 0.0595
```

```
[ ] # 10: Presence of Exclamation Marks
    df['exclamation_count'] = df['message'].apply(lambda x: x.count('!'))

    print(f"Average ! in spam: {df[df['label'] == 'spam']['exclamation_count'].mean():.2f}")
    print(f"Average ! in ham: {df[df['label'] == 'ham']['exclamation_count'].mean():.2f}")
```

```
Average ! in spam: 0.73
Average ! in ham: 0.17
```

```
[ ] #11: Presence of Question Marks
    df['question_count'] = df['message'].apply(lambda x: x.count('?'))

    print(f"Average ? in spam: {df[df['label'] == 'spam']['question_count'].mean():.2f}")
    print(f"Average ? in ham: {df[df['label'] == 'ham']['question_count'].mean():.2f}")
```

```
Average ? in spam: 0.23
Average ? in ham: 0.28
```

```
[ ] #12: Word Length Distribution
    def avg_word_length(text):
        words = text.split()
        return sum(len(word) for word in words) / len(words) if len(words) > 0 else 0

    df['avg_word_length'] = df['message'].apply(avg_word_length)

    print(f"Average word length in spam: {df[df['label'] == 'spam']['avg_word_length'].mean():.2f}")
    print(f"Average word length in ham: {df[df['label'] == 'ham']['avg_word_length'].mean():.2f}")
```

```
Average word length in spam: 4.99
Average word length in ham: 4.18
```

```
[ ] # 13: Presence of Urgency Words
    urgency_words = ['urgent', 'now', 'immediately', 'hurry', 'quick', 'last chance']
    df['urgency_count'] = df['message'].apply(lambda x: sum(x.lower().count(word) for word in urgency_words))

    print(f"Average urgency words in spam: {df[df['label'] == 'spam']['urgency_count'].mean():.2f}")
    print(f"Average urgency words in ham: {df[df['label'] == 'ham']['urgency_count'].mean():.2f}")
```

```
Average urgency words in spam: 0.40
Average urgency words in ham: 0.13
```

```
[ ] #14:Message Start Analysis
    spam_starts = df[df['label'] == 'spam']['message'].str[:10].value_counts().head(5)
    ham_starts = df[df['label'] == 'ham']['message'].str[:10].value_counts().head(5)

    print("Common starts in spam messages:")
    print(spam_starts)
    print("\nCommon starts in ham messages:")
    print(ham_starts)
```

```
Common starts in spam messages:
message
URGENT! Yo    17
PRIVATE! Y    16
Congratula    11
Do you wan    10
URGENT! We    10
Name: count, dtype: int64

Common starts in ham messages:
message
Sorry, I'l    38
Good after    14
I cant pic    12
I want to     11
How are yo    11
Name: count, dtype: int64
```

```
[ ] #15:Presence of Special Characters
    special_chars = r'[^\w\s]'
    df['special_count'] = df['message'].str.count(special_chars)

    print(f"Average special chars in spam: {df[df['label'] == 'spam']['special_count'].mean():.2f}")
    print(f"Average special chars in ham: {df[df['label'] == 'ham']['special_count'].mean():.2f}")
```

```
Average special chars in spam: 6.15
Average special chars in ham: 3.97
```

```
[ ] #16: Time References in Messages
    time_pattern = r'\d{1,2}(?:am|pm|AM|PM|:\d{2})?'
    df['time_ref_count'] = df['message'].str.count(time_pattern)

    print(f"Average time references in spam: {df[df['label'] == 'spam']['time_ref_count'].mean():.2f}")
    print(f"Average time references in ham: {df[df['label'] == 'ham']['time_ref_count'].mean():.2f}")
```

```
Average time references in spam: 9.44
Average time references in ham: 0.27
```

```
[ ] #17: Presence of Personal Pronouns
    pronouns = ['I', 'you', 'he', 'she', 'we', 'they', 'me', 'him', 'her', 'us', 'them']
    df['pronoun_count'] = df['message'].apply(lambda x: sum(x.lower().count(pronoun.lower()) for pronoun in pronouns))

    print(f"Average pronouns in spam: {df[df['label'] == 'spam']['pronoun_count'].mean():.2f}")
    print(f"Average pronouns in ham: {df[df['label'] == 'ham']['pronoun_count'].mean():.2f}")
```

```
Average pronouns in spam: 7.92
Average pronouns in ham: 6.34
```

```
[ ] #18:: Presence of Numeric Digits
    df['digit_count'] = df['message'].str.count(r'\d')

    print(f"Average digits in spam: {df[df['label'] == 'spam']['digit_count'].mean():.2f}")
    print(f"Average digits in ham: {df[df['label'] == 'ham']['digit_count'].mean():.2f}")
```

```
Average digits in spam: 15.76
Average digits in ham: 0.30
```

```
[ ] #19: Message-Level Analysis
    df = df.rename(columns={'v1': 'label', 'v2': 'message'})[['label', 'message']]
    df['message_length'] = df['message'].apply(len)
    df['word_count'] = df['message'].apply(lambda x: len(x.split()))

    print(df.head())
```

```
  label                                            message  message_length  \
0   ham  Go until jurong point, crazy.. Available only ...             111
1   ham                      Ok lar... Joking wif u oni...              29
2  spam  Free entry in 2 a wkly comp to win FA Cup fina...             155
3   ham  U dun say so early hor... U c already then say...              49
4   ham  Nah I don't think he goes to usf, he lives aro...              61

   word_count
0          20
1           6
2          28
3          11
4          13
```

```
[ ] #20: Type-Level Aggregation

    grain2 = df.groupby('label').agg({
        'message': 'count',
        'message_length': 'mean',
        'word_count': 'mean'
    }).reset_index()

    grain2 = grain2.rename(columns={
        'message': 'message_count',
        'message_length': 'avg_message_length',
        'word_count': 'avg_word_count'
    })

    print(grain2)
```

```
  label  message_count  avg_message_length  avg_word_count
0   ham           4825           71.023627       14.200622
1  spam            747          138.866131       23.851406
```