Name : Ayush N. shinde
Batch : E-12 Roll no ; ETI-35
PRN NO ; 202401070112.

D D M M Y Y Y Y

Theory Activity No. 1

20 Problems statements and solutions.
(SMS Spam Collection)

Below are 20 problem statements covering various analyse (counting, filtering, grouping, statics) on the dataset. Each problem is solved using Pandas and NumPy.

P-1: How many SMS messages are in the dataset ?
→
```
total_messages = len(df)
print(F"Total SMS messages : {total_messages}")
```

Expl : Pandas' len() counts the number of rows (messages) in the DataFrame.

P-2: How many messages are spam and how many are ham ?
→
```
label_counts = df['label'].value_counts()
print("Spam and Ham counts:\n", label_counts)
```

Expl : value_counts() groups and counts occurrences of each label (spam/ham).

P-3: What is the percentage of spam messages ?
→
```
spam_percentage = (df['label'] == 'spam').mean() * 100
print(F"percentage of spam messages : {spam_percentage:.2F} %")
```

Expl: mean() computes the proportion of spam messages, multiplied by 100 for percentage.

P-4: calculate the length of each message and add it as a new column

→
```
df['message_length'] = df['message'].apply(len)
print(df[['message', 'message_length']].head())
```

Expl: apply(len) computes the character count of each message, stored in a new column.

P-5: What is the average message length for spam and ham messages?

→
```
avg_length = df.groupby('label')['message_length'].mean()
print("Average message length by label:\n", avg_length)
```

Expl: groupby() and mean() calculate the average message_length for each label.

P-6: What are the lengths of the longest and shortest messages?

→
```
max_length = df['message_length'].max()
min_length = df['message_length'].min()
print(f"longest message length: {max_length},
      shortest message length: {min_length}")
```

Expl: max() & min() find the extreme values in message_length.

D D M M Y Y Y Y

P-7: Which messages are longer than 150 characters?

→
```
long_messages = df[df['message_length'] > 150]
                [['label', 'message', 'message_length']]
print("Messages longer than 150 characters:\n",
      long_messages.head())
```

Expl: Boolean indexing filters messages with messages with message_length > 150

P-8: How many unique words are in the entire dataset?

→
```
all_words = ' '.join(df['message']).split()
unique_words = len(np.unique(all_words))
print(f"Number of unique words: {unique_words}")
```

Expl: Joins all messages, splits into words, and uses NumPy's unique() to count distinct words.

P-9: What is the most common word in spam messages?

→
```
spam_words = ' '.join(df[df['label'] == 'spam']['
             message']).split()
Word_counts = pd.series(spam_words).value_
                        counts()
most_common = word_counts.idxmax()
print(f"most common word in spam: {
                most_common}").
```

Expl: filters spam messages, splits into words, and uses value_counts() to find the most frequent word.

D D M M Y Y Y Y

P-10: What is the standard deviation of message lengths for spam messages?

→
```
spam_std = np.std(df[df['label'] == 'spam']['message_length'], ddof = 1)
print(f" standard deviation of spam message lengths : {spam_std: .2f}")
```

Expl: NumPy's std() computes the standard deviation for spam message lengths.

P-11: Which messages contain the word "Free"?

→
```
Free_messages = df[df['message'].str.contains('Free', case = False, na = False)][['label', 'message']]
print("messages containing 'Free':\n", Free_messages.head())
```

Expl: str.contains() filters messages with "free" (case-insensitive).

P-12: What is the average number of words in spam vs. ham messages?

→
```
df['word_count'] = df['message'].apply(lamda x: len(x.split()))
avg_words = df.groupby('label')['word_count'].mean()
print("Average word count by label:\n", avg_words).
```

Expl: Adds word_count column by splitting messages into words, then computes average per label.

D D M M Y Y Y Y

P-13: What are the 5 longest spam messages?

→
```
top_spam = df[df['label'] == 'spam'][['message', 'message_length']].nlargest(5, 'message_length')
print("Top 5 longest spam messages:\n", top_spam)
```

Expl: nlargest() retrieves the top 5 spam messages by message_length.

P-14: What is the correlation between message length and word count?

→
```
correlation = np.corrcoef(df['message_length'], df['word_count'])[0,1]
Print(f"correlation between message length and word count: {correlation:.3f}")
```

Expl: NumPy's corrcoe() calculates the pearson correlation between message_length and word_count.

P-15: How many messages have more than 10 words?

→
```
many_words = (df['words_count'] > 10).sum()
print(f"Messages with more than 10 words: {many_words}")
```

Expl: count messages with word_count > 10 using boolean indexing and sum().

P-16: What proportion of spam message contain "win"?

→
```
spam_win = df[df['label'] == 'spam']['message'].
    str.contains('win', case = False, na = False).
                                    mean()
print(f"proportion of spam message with 'win':
                        {spam_win:.3f}")
```

Expl: filters spam messages, checks for "win" with str.contains(), and computes proportion with mean()

P-17: what is the median message length for spam and ham messages?

→ median_length = df.groupby() and median() compute the median message_length for each label.

```
Median_length = df.groupby('label')['message'_
                    length'].median()
print("median message length by label:\n",
                            median_length)
```

P-18 which messages are duplicates?

→
```
duplicates = df[df['message'].duplicated()][['label'
                                    message']]
print("Duplicate messages:\n", duplicates.head())
```

Expl: duplicated() identifies messages with repeated message values.

D D M M Y Y Y Y

P-19: What is the interquartile range of message lengths for ham messages?

→
```
ham_lengths = df[df['label'] == 'ham']['message_length']
iqr = np.percentile(ham_lengths, 75) - np.percentile(ham_lengths, 25)
print(f"IQR of ham message lengths: {iqr}")
```

Expl: Numpy's percentile() computes the IQR (Q3 - Q1) for ham message lengths.

P-20: create a pivot table showing average message length by label.

→
```
pivot = pd.pivot_table(df, values = 'message_length', index = 'label', aggFunc = 'mean')
print("Pivot table of average message length by label:\n", pivot)
```

Expl: pivot_table() computes the mean message_length for each label.

Notes :

- Dataset Assumption : I assumed the dataset has label and message columns. If your dataset has different columns or format.
- Requirements : Ensure pandas and numpy are installed (pip install pandas numpy). if using a local file, update the file path in Pd.read_csv()
- output : The outputs shown are illustrative. Actual results depend on the dataset's content.