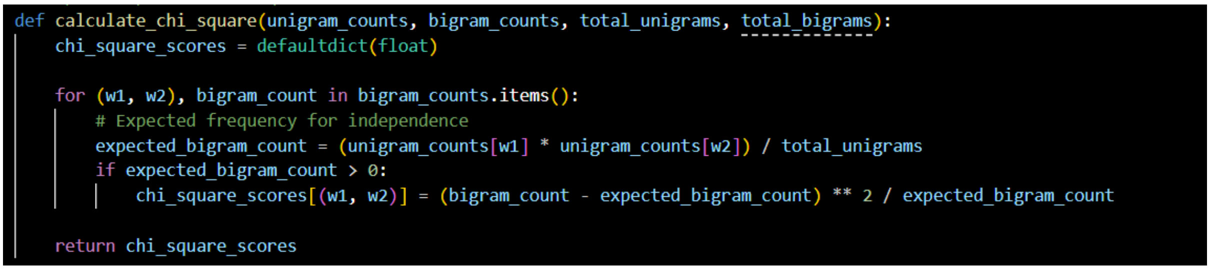
Collocation answers

Divya Trichy Ravi

● How complete your program is. Even if your program is not complete or you are getting compilation errors, you will get partial credit proportionally. Just mention clearly and accurately how far you got.

1. The program was constructed using the given clear instruction.
2. The collocations.py contains two user-defined functions pmi and chi-square.
3. These two functions read the data, process it and extract the core features and own features.
4. To run this program download the zip file SBD_HW1 and extract it.
5. Place it in a respective directory and open command prompt inside that directory.
6. Using this command python collocations.py colloctions.txt.
7. Here the two different functions
    a. PMI – Pointwise Mutual Information
    b. Chi-square

● If your program is complete, the top 20 bigrams along with their chi-square scores, in reverse order of the scores.

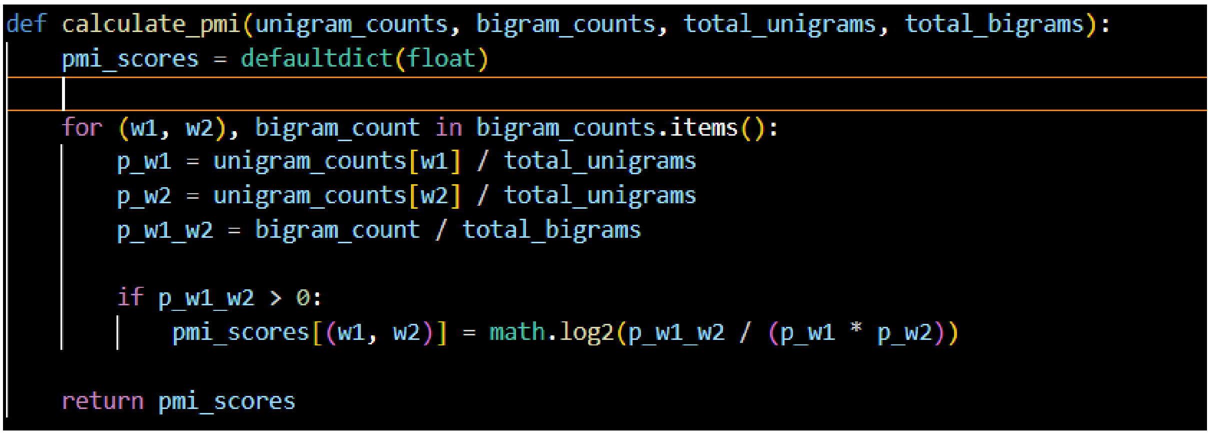


```
def calculate_chi_square(unigram_counts, bigram_counts, total_unigrams, total_bigrams):
    chi_square_scores = defaultdict(float)

    for (w1, w2), bigram_count in bigram_counts.items():
        # Expected frequency for independence
        expected_bigram_count = (unigram_counts[w1] * unigram_counts[w2]) / total_unigrams
        if expected_bigram_count > 0:
            chi_square_scores[(w1, w2)] = (bigram_count - expected_bigram_count) ** 2 / expected_bigram_count

    return chi_square_scores
```

Bigram	Chi – Square Values
Golf Course	428743.0000
multi-spired castle-like	428743.0000
Traveling Abroad	428743.0000
Three-month T-bill	428743.0000
Competing asset-allocation	428743.0000
Mill Playhouse	428743.0000
Perfect Witness	428743.0000
Witness Aidan	428743.0000
Stockard Channing	428743.0000
Final Days	428743.0000
Brideshead Revisited	428743.0000
schizoid horror	428743.0000

Performances kicks	428743.0000
Polished hooves	428743.0000
Shiny Nikes	428743.0000
Glory Enough	428743.0000
Ku Klux	428743.0000
Klux Klan	428743.0000
Latest Period	428743.0000
slow-growth low-profit	428743.0000

- **If your program is complete, the top 20 bigrams along with their PMI scores, in reverse order of the scores.**



```
def calculate_pmi(unigram_counts, bigram_counts, total_unigrams, total_bigrams):
    pmi_scores = defaultdict(float)

    for (w1, w2), bigram_count in bigram_counts.items():
        p_w1 = unigram_counts[w1] / total_unigrams
        p_w2 = unigram_counts[w2] / total_unigrams
        p_w1_w2 = bigram_count / total_bigrams

        if p_w1_w2 > 0:
            pmi_scores[(w1, w2)] = math.log2(p_w1_w2 / (p_w1 * p_w2))

    return pmi_scores
```

Bigram	**PMI -Values**
Golf Course	18.7098
multi-spired castle-like	18.7098
Traveling Abroad	18.7098
Three-month T-bill	18.7098
Competing asset-allocation	18.7098
Mill Playhouse	18.7098
Perfect Witness	18.7098
Witness Aidan	18.7098
Stockard Channing	18.7098
Final Days	18.7098
Brideshead Revisited	18.7098
schizoid horror	18.7098
Performances kicks	18.7098
Polished hooves	18.7098
Shiny Nikes	18.7098
Glory Enough	18.7098

Ku Klux	18.7098
Klux Klan	18.7098
Latest Period	18.7098
slow-growth low-profit	18.7098

● **A brief discussion of which of the two measures you believe works better to identify collocations, based on your analysis of the top 20 bigrams produced by each measure.**

Therefore, when it comes to determining the effectiveness of Chi-square and PMI in identifying collocations within big datasets, Chi-square is somewhat better suited for handling this. The strong robustness that it possesses in handling large sizes means its test of independence is ideally more effective in bringing on board crucial word associations that cut across in diverse contexts. Chi-square is different from PMI because the latter is sensitive to data sparsity and frequently occurring common words, which tend to give misleading results; Chi-square provides meaningful outcomes without easily being distorted by the size of the dataset. Therefore, the Chi-square method is the favorite option when it comes to large datasets where reliable statistical analysis is necessary in the search for truly correct collocations.