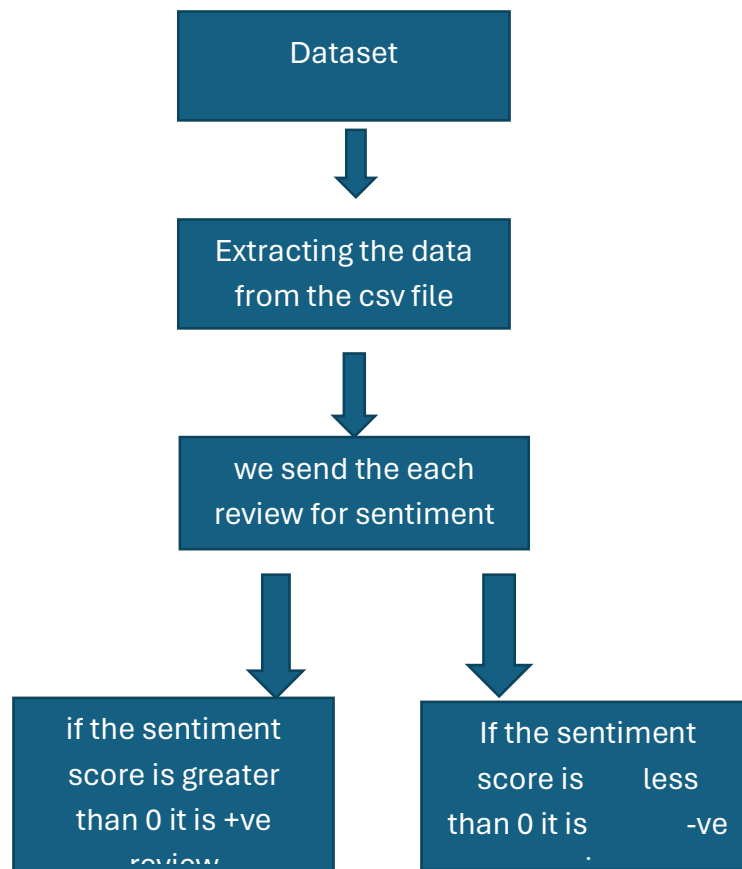Approach and Results of Sentiment analysis.

1)Resource based Classification:

It is the basic approach, where the words get the polarity and we add positive polarity score and negative polarity score, we add then together for sentence and we by the result of the sum of polarities of each word will determine it is a good review or bad review.

```
          ┌─────────────────┐
          │     Dataset     │
          └────────┬────────┘
                   │
                   ▼
          ┌─────────────────┐
          │ Extracting the  │
          │ data from the   │
          │    csv file     │
          └────────┬────────┘
                   │
                   ▼
          ┌─────────────────┐
          │  we send the    │
          │ each review for │
          │   sentiment     │
          └──────┬───────┬──┘
                 │       │
          ┌──────▼──┐ ┌──▼──────┐
          │ if the  │ │ If the  │
          │sentiment│ │sentiment│
          │score is │ │score is │
          │greater  │ │  less   │
          │than 0 it│ │than 0 it│
          │is +ve   │ │is  -ve  │
          │review   │ │         │
          └─────────┘ └─────────┘
```

To get the sentiment score:

WE TOKENIZE the sentence into words. Then for each word in that list will get a sentiment score by **sentiword.net.**

After the sentiment analysis we write, review with sentiment(good/bad) in a new csv file named as result_of_sentiment.csv.

```python
4    from nltk.tokenize import word_tokenize
5
6    # Download required NLTK resources
7    nltk.download('punkt')
8    nltk.download('wordnet')
9    nltk.download('sentiwordnet')
10   nltk.download('averaged_perceptron_tagger')
11
12   # # Function to calculate sentiment score for a word
13   def calculate_word_sentiment_score(word, pos_tag):
14       synsets = list(swn.senti_synsets(word, pos_tag))
15       if synsets:
16           # Use the average of all available synsets
17           score = sum([synset.pos_score() - synset.neg_score() for synset in
18           return score
19       else:
20           return 0
```

Files: resource_based_classification.py, results_of_sentiment.csv, sentiment_by_rating.py.py, tripadvisor_hotel_reviews.csv

```
Review,Sentiment
"nice hotel expensive parking got good deal stay hotel anniversary, arrived late evening took advice previous reviews did valet parking,
"ok nothing special charge diamond member hilton decided chain shot 20th anniversary seattle, start booked suite paid extra website descr
"nice rooms not 4* experience hotel monaco seattle good hotel n't 4* level.positives large bathroom mediterranean suite comfortable bed
```

Using Classifiers:

I had done with 5 classifers to find out the which one is with the best accuracy. Attaching the screen shots for each classifer results.

```
Metrics for Logistic Regression Model
Accuracy: 0.9516955354964626
```

```
Metrics for Decision Tree
Accuracy: 0.9004635276896804
```

```
Metrics for Voting Classifier
Accuracy: 0.9011954135154916
```

```
Metrics for  Support Vector Machine
Accuracy: 0.9458404488899732
```

```
Metrics for  Multi-Nomibal Naive Bayes
Accuracy: 0.928275189070505
```