Word2Vec.

Word Embeddings:-

- Represent words as vectors of numbers.
- Useful for creating similarity b/w words.
- Captures semantic meaning through vector operations.

Cosine Similarity

- Measures similarity b/w two vectors.
- Formula:-

$$\text{Cosine similarity} = \frac{A \cdot B}{\|A\| \times \|B\|}$$

- Vectors w/ high cosine similarity point in the same direction

Skip Gram Model:-

- Predicts surrounding words (content) given a target word.
- Efficient for learning word embeddings.
- Training involves maximizing the prob of content words given a target word.

- Negative Sampling:-
  - Improves training efficiency by using logistic regression model.
  - Introduces negative examples (words not in context) to prevent trivial solutions.

- Formula for the model's prediction:-

$$\text{Prediction} = \sigma(V_{input} \cdot V_{content})$$

where σ is the sigmoid func.

Training Process:-

- Initialize embeddings & content matrics randomly.
- Use dot product & sigmoid to predict if words are neighbors.
- Calculate error:-

$$\text{Error} = \text{Target} - \text{Prediction}.$$

- Adjust embeddings to minimize error.
- Important Hyperparameter:-

→ Window Size:- Smaller (2-15) captures words similarity, larger (15-50) captures relatedness.

Number of negative sample:- Typically

5-20, 2-5 for large datasets.

Analogy Operation:-

- Example :- "King" - "Man" + "Woman" ≈ "Queen".
- Demonstrate vector arithmetic in embedding space.