Before realizing how TF-IDF adds nuance by weighting terms by rarity, I saw the text as just word counts. Later, I understood how embeddings embed semantic relationships in dense vectors. The most unexpected realization was how linear aritBag-of-Words (BOW), TF-IDF, n-grams, and word embeddings are the four different approaches I investigated in Lab 04 to transform unprocessed text into numerical features appropriate for machine learning. Each approach has its trade-offs in terms of simplicity, interpretability, sparsity, and semantic richness, and each represents a distinct philosophy about the most significant elements of language. Semantic and syntactic links can be represented as consistent vector offsets when hmetic in embedding space (e.g., "king" – "man" + "woman" ≈ "queen") is used.

First of all, creating Bag-of-Words from scratch made me realize how simple and practical it can be. I was able to create a document-term matrix and train a Naive Bayes classifier with respectable baseline performance by merely counting the number of times each stemmed, stop-word-filtered token occurs in a document. The primary benefit of BOW is its transparency: each feature is explicitly associated with a particular word, making it possible to determine which phrases influence a model's judgments by looking at the coefficients. The drawback is that BOW disregards word order and context and handles each token separately. The resulting matrix is very sparse—more than 90% of entries are zero—making it ineffective for huge vocabularies. This is because sentences with very different meanings—for example, "dog chased cat" vs. "cat chased dog"—collapse into identical vectors.

Next, we went to TF-IDF, which reweights each phrase according to its rarity across documents. This involved emphasizing discriminative phrases like "fantastic" or "boring" and down-weighting common words like "movie" and "acting" in our five-review toy dataset. The accuracy of the same Multinomial Naive Bayes model increased by about 10 percentage points when I retrained it using TF-IDF vectors. Documents that share uncommon but instructive tokens wind up closer in the feature space, confirming that TF-IDF frequently produces higher relevant similarity scores and classification limits. Although TF-IDF still has sparsity issues, it generates clearer signals for class distinction.

I then examined trigrams, bigrams, and unigrams to get short-range context. The creation of n-grams exposed phrases that a unigram model is unable to discern, such as "not recommended" and "absolutely fantastic." Trigrams allowed me to see multi-word praise patterns ("excellent acting performance"), while Bigrams were especially good in indicating sentiment flips ("not recommended," "quite boring"). The trade-off is obvious: as n rises, the number of n-grams that can be created skyrockets, making the matrix even more sparse and needing a lot more computation and memory. Using count criteria or restricting to bigrams is a reasonable compromise in practice.

Lastly, I investigated dense, low-dimensional representations by diving into pre-trained GloVe embeddings. I created a similarity matrix and reported the top five most comparable neighbors for each keyword after filtering a list of movie-related terms. Semantic clusters were interesting to observe: "good" and "bad" occupied opposite ends of a feeling axis, "movie" and "film" were nearly interchangeable, and "actor" and "director" sat close to one another. After witnessing personally how vector arithmetic encodes consistent semantic transformations, reading about comparisons like "king – man + woman ≈ queen" made more sense. Embeddings are now dense vectors that capture context, meaning, and even small biases passed down from their training corpora, rather than sparse bags of token counts.

All things considered, my classification tests demonstrated that TF-IDF outperformed raw BOW and n-gram models, striking the ideal balance between simplicity and discriminative ability in this tiny dataset. Although the embedding-based features were not tested end-to-end in this lab, contextualized models (like BERT) or embeddings will probably do better in capturing subtle sentiment or topic differences in a bigger corpus.

My understanding of the range of text representations has grown as a result of thinking about these techniques. By the end of the lab, I realized that weighting schemes and vector geometries might encode semantics, word order, and hierarchical information, whereas at the beginning I thought of text as just word counters. The most unexpected realization was how much information is contained in the relative locations of dense vectors and how an algebraic combination of these vectors might uncover parallels. In order to create more equitable NLP systems, I'm excited to investigate contextual embeddings, transformer-based classifiers, and word vector debiasing methods in the future. A strong basis for the transition from basic baselines to cutting-edge text representations has been established by this lab.