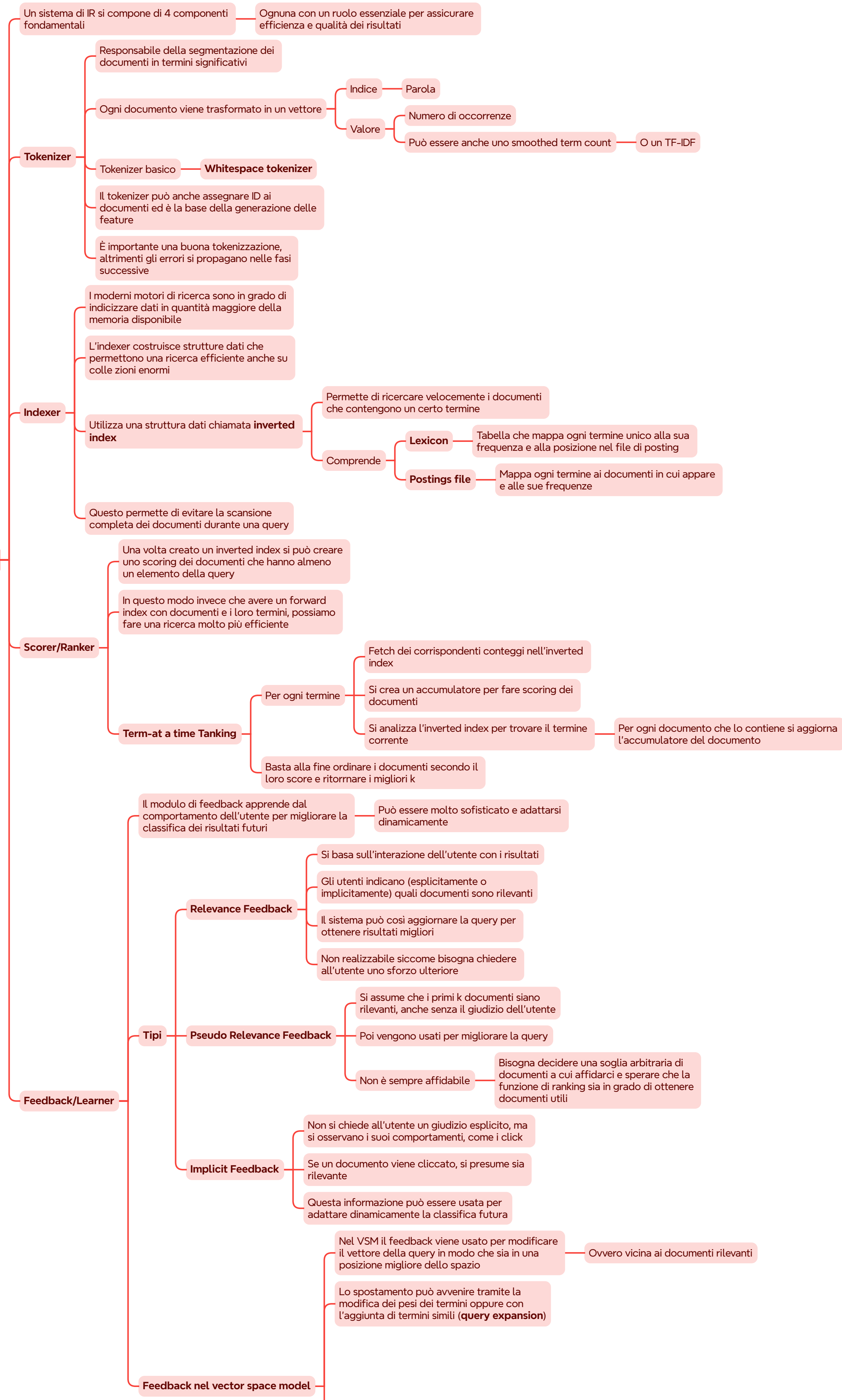


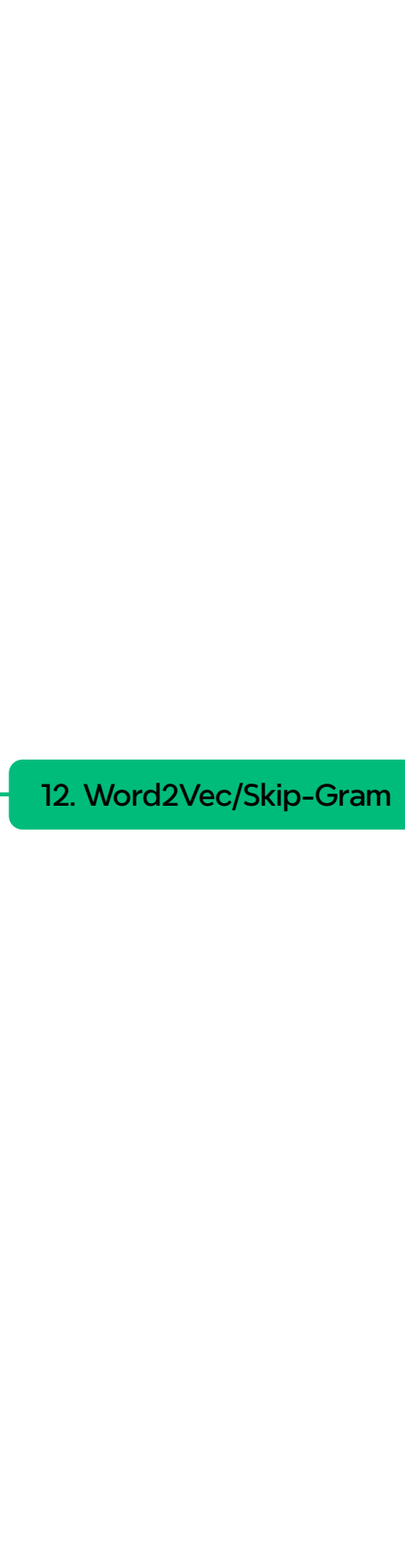
Text Analysis (Parte 3)

9. Search Engine Implementation

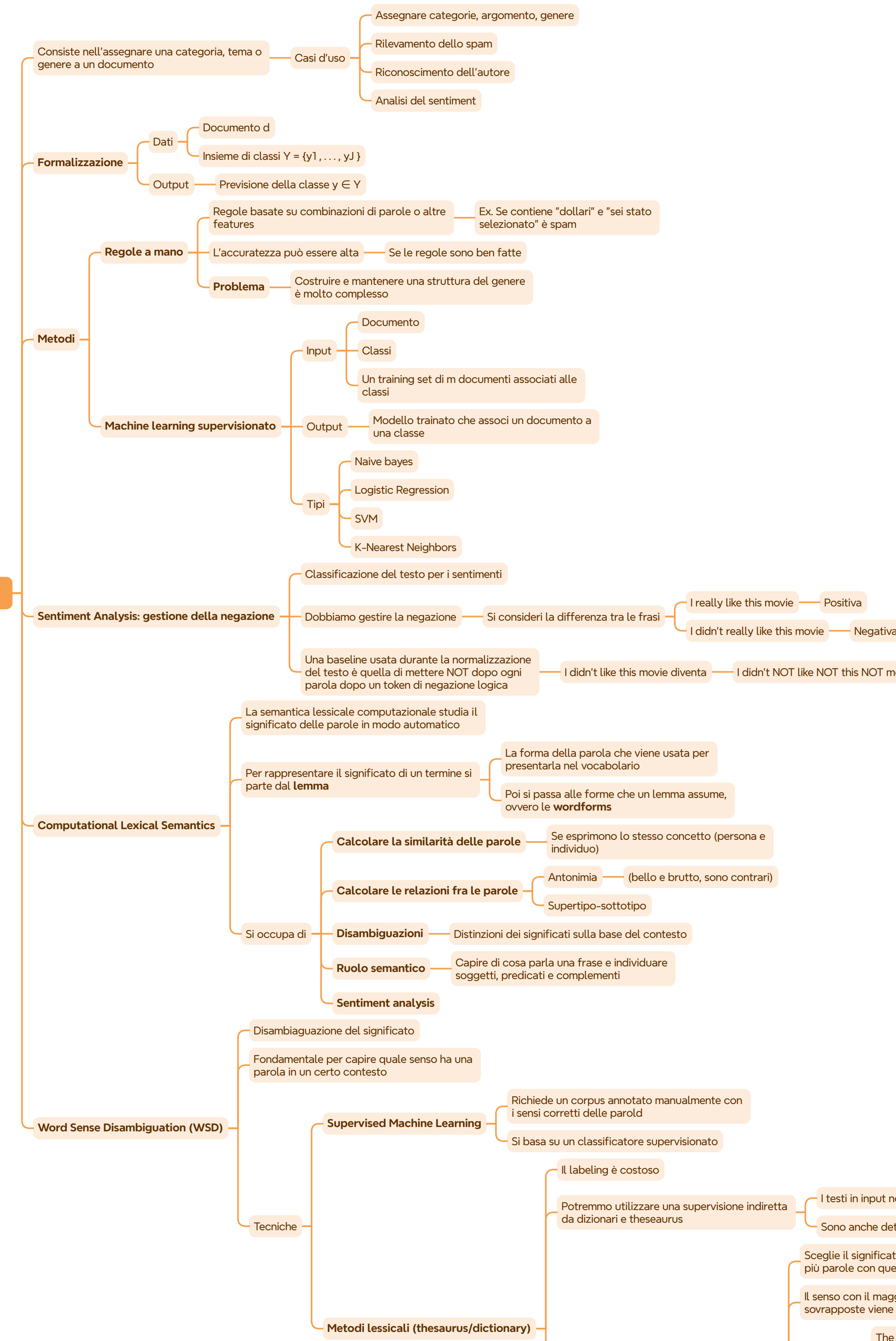


11. Vector Semantics and Embeddings

12. Word2Vec/Skip-Gram



10. Text Classification



$$\hat{q}_m = \alpha \hat{q} + \beta \sum_{d_i \in D_r} \hat{d}_i - \gamma \sum_{d_i \in D_n} \hat{d}_i$$

La nuova query qm viene calcolata come

- Query originale
- Insieme dei documenti non rilevanti
- Insieme dei documenti non rilevanti
- Peso che controlla i termini della query originale
- Peso che avvicina la query al centroide dei termini dei documenti rilevanti
- Peso che allontana la query dal centroide dei termini dei documenti non rilevanti
- Si vuole polarizzare la query verso i risultati rilevanti
- β > γ
- È comune ridurre γ o escludere del tutto i documenti non rilevanti per evitare "tunnel" nel nuovo vettore di query

$$PMI(w, c) = \log_2 \frac{P(w, c)}{P(w)P(c)}$$

Nel nostro caso, probabilità che la parola w sia nel contesto di un'altra parola tratto la probabilità che la due parole compaiano nel documento, senza alcuna relazione tra le due

$$PPMI(w, c) = \max(\log_2 \frac{P(w, c)}{P(w)P(c)}, 0)$$

Si fa uso della Positive PMI

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

SVD scompone la matrice in input in tre matrici