CV Ranking Project Report

**Marwan Ghazal – 231000765**

**Omar Amin – 231000573**

**Mohamed Abdellatif – 231000119**

**Zeyad Ahmed – 231000010**

**Project Overview**

This project aims to build an intelligent system that ranks CVs (resumes) based on their relevance to a job description, using historical acceptance decisions. The core objective is to train machine learning models that can predict the likelihood of a candidate being accepted for a given job, and then rank candidates accordingly.

**Data Extraction**

## The dataset was sourced from Kaggle and consists of three main columns:

- resume\_Text: textual content of applicant resumes.

- job\_description\_text: job posting descriptions.

- decision: binary value indicating if a candidate was accepted (1) or denied (0).

Each job listing is associated with multiple CVs, creating a many-to-one relationship. The dataset is textual and requires extensive preprocessing before modeling.

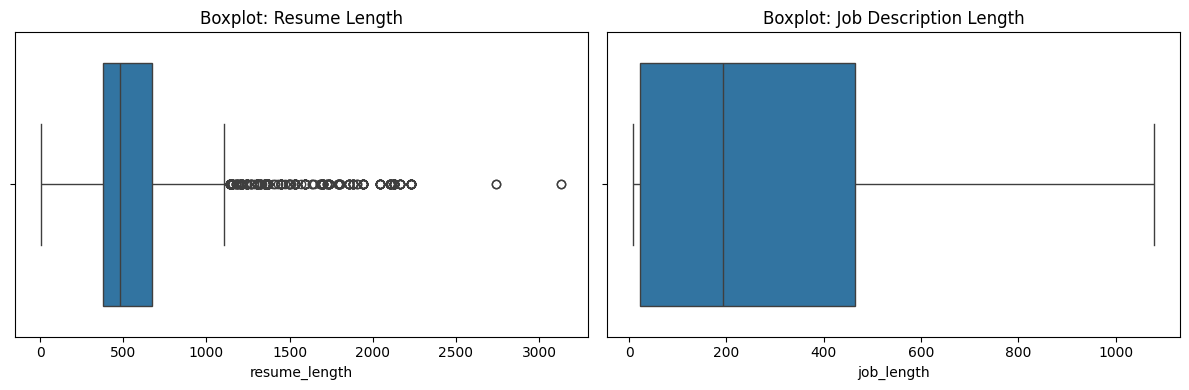
**Data Preprocessing**

## Several cleaning and filtering steps were applied to prepare the data:

1. Job Filtering: Jobs with fewer than 10 CVs were excluded to ensure sufficient data per job for learning patterns.

2. Class Balance Filtering: Jobs where all resumes were either accepted or denied were removed to ensure the model could learn decision boundaries.

3. Outlier Detection: Resume and job description length distributions were visualized to identify outliers.



4. Re-indexing: After filtering, job\_id values were re-encoded to be contiguous for easier tracking.

**Encoding Techniques**

## Three main encoding strategies were used to convert the text data into numerical features:

1. TF-IDF (Term Frequency-Inverse Document Frequency)

- Converts text into sparse matrix of weighted word counts.

- Captures the importance of words across documents.

- Fast and interpretable.

2. BERT (Bidirectional Encoder Representations from Transformers)

- Uses pre-trained transformer models to generate semantic embeddings.

- Captures deep contextual relationships.

- Ideal for transfer learning with rich language understanding.

3. spaCy Embeddings

- Uses en\_core\_web\_md to generate fixed-length vectors from resumes and job descriptions.

- Faster than BERT, lighter semantic coverage.

**Modeling**

Various models were applied to each encoding method.

## Traditional Models:

- Logistic Regression: Predicts the probability of a binary outcome using a linear function.

- Random Forest: Builds multiple decision trees and averages their predictions.

- LinearSVC: Finds the optimal hyperplane that separates classes in high-dimensional space.

- XGBoost: Gradient boosting algorithm that builds trees sequentially, minimizing error.

- LightGBM: A fast gradient boosting framework that uses histogram-based splitting.

- Naive Bayes (Gaussian): assuming features are independent and normally distributed.

- K-Nearest Neighbors (KNN): Classifies based on the majority class among the k closest data points.

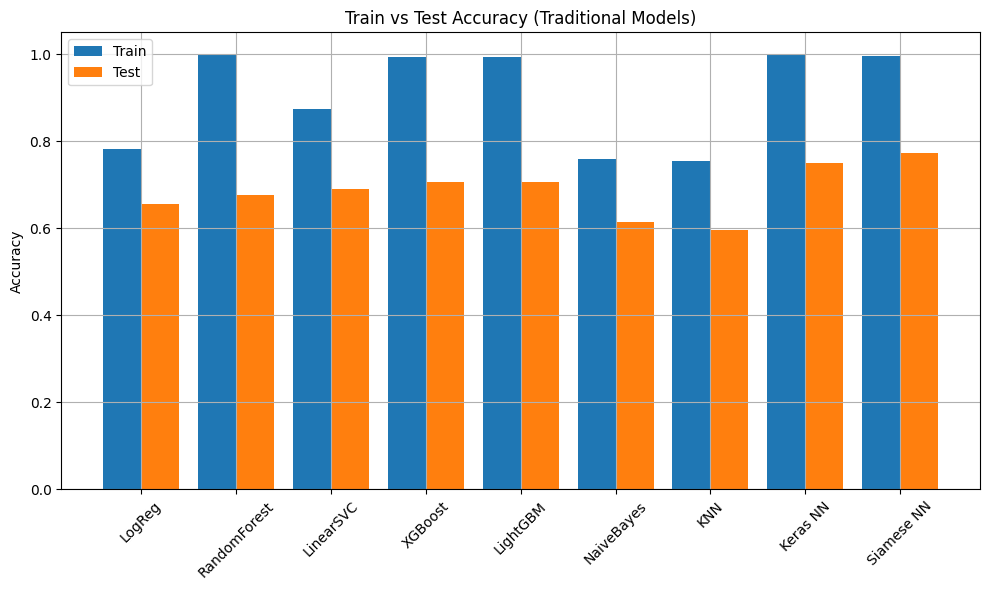
## Neural Networks:

- Keras Fully Connected Neural Network (NN)

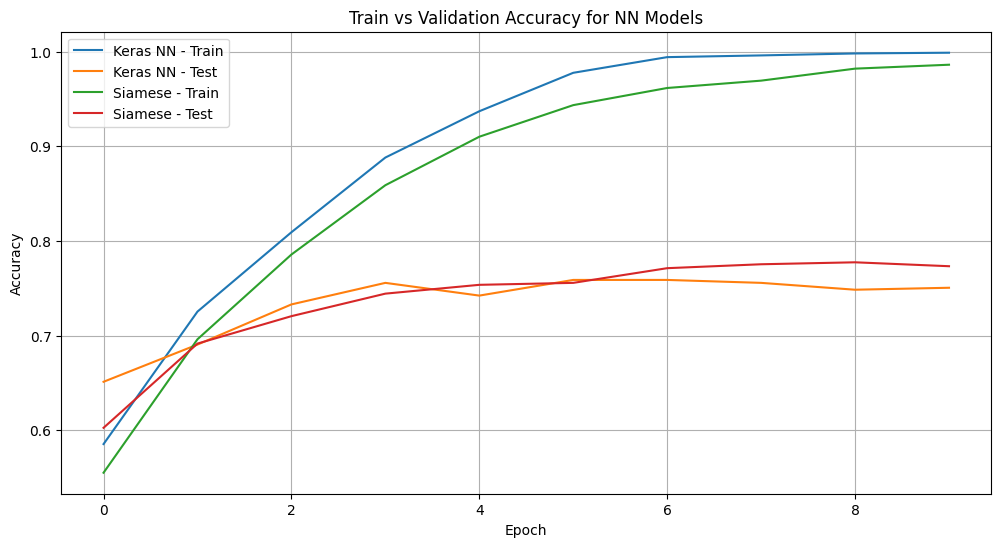
- Siamese Neural Network

## TF-IDF Results:

Accuracy Comparison:



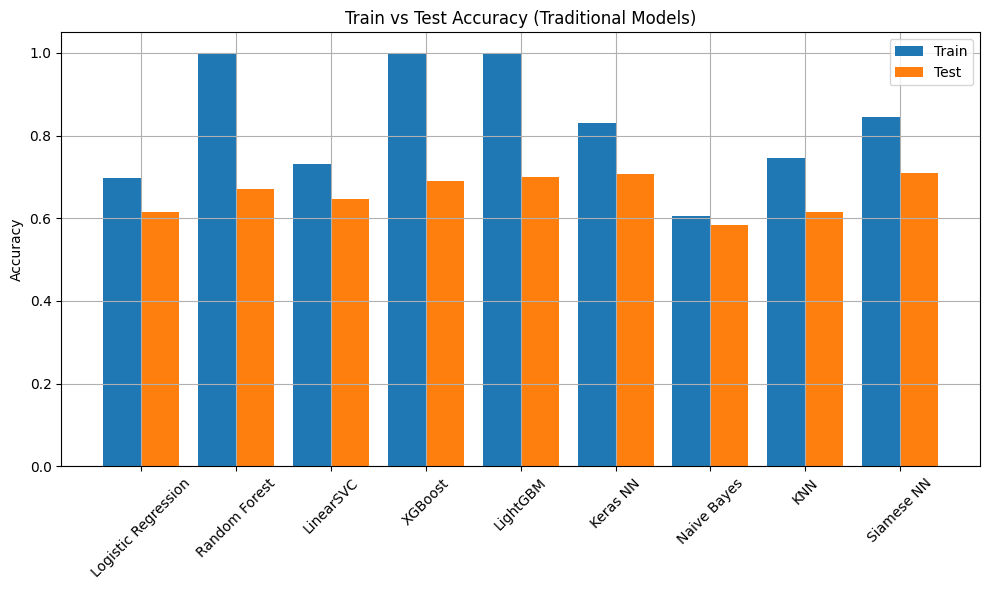
Keras and Siamese NN Accuracy over Epochs:



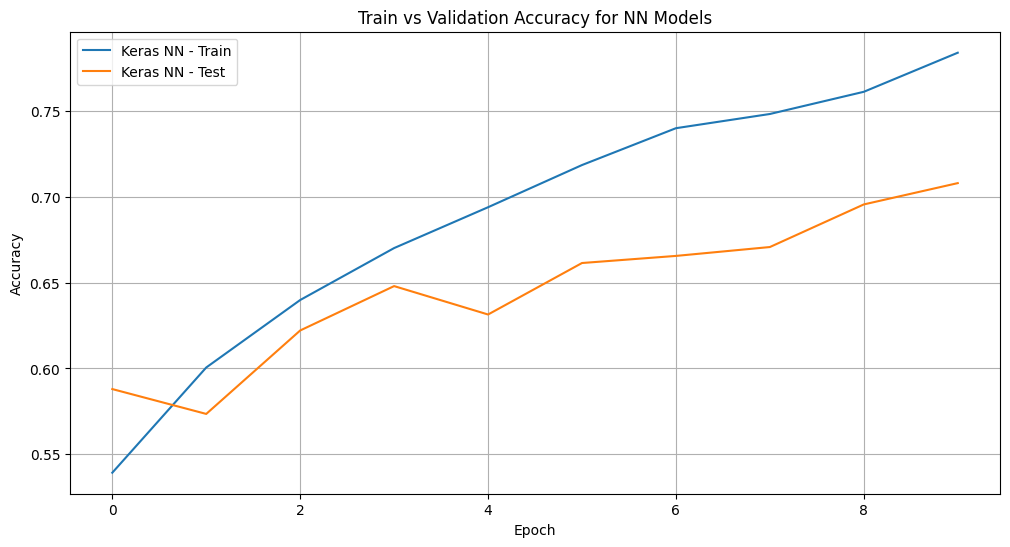
Best TF-IDF Model: Logistic Regression Network with ~66% accuracy.

## BERT Results:

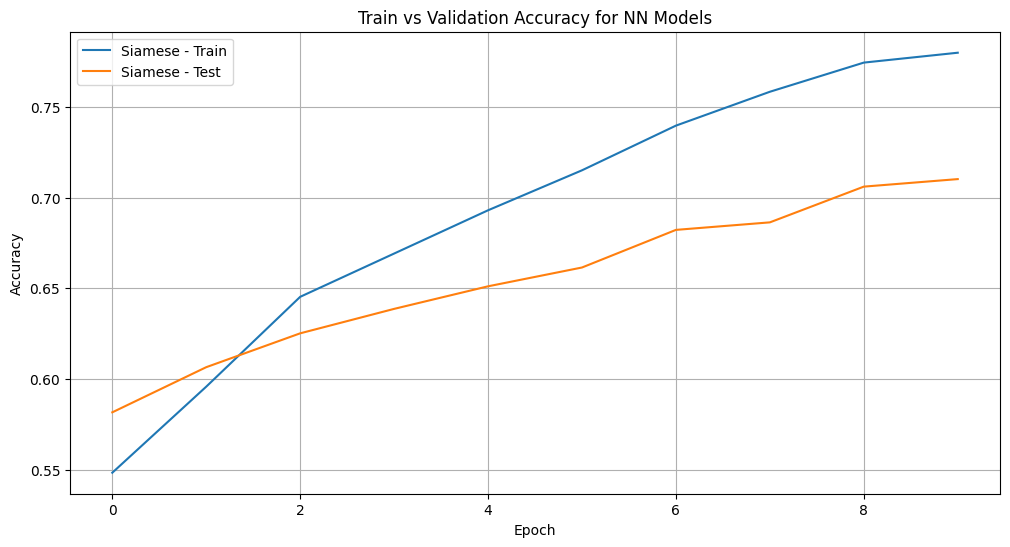
Accuracy Comparison:



Keras NN Accuracy over Epochs:



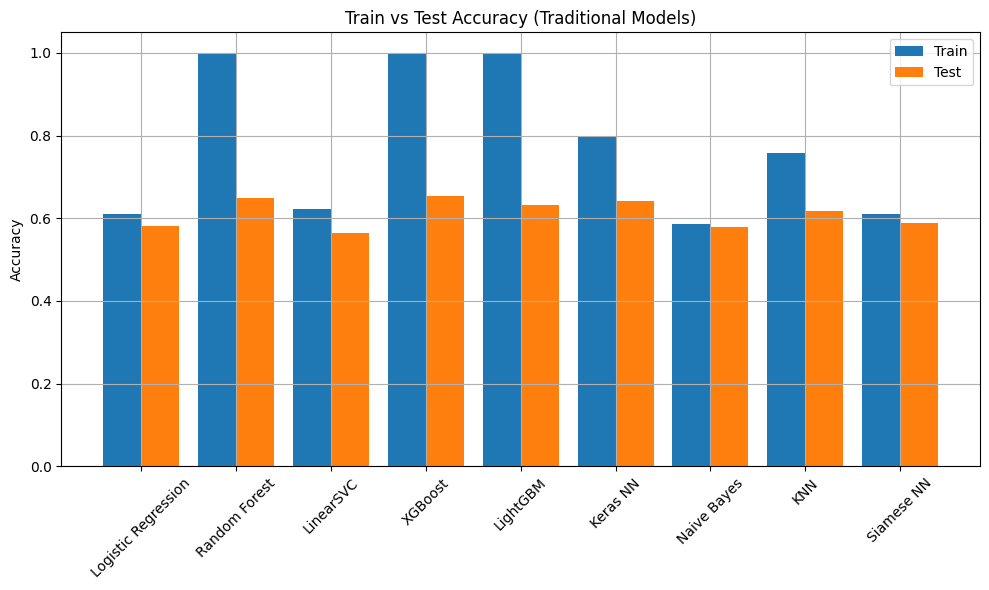
Siamese Accuracy:



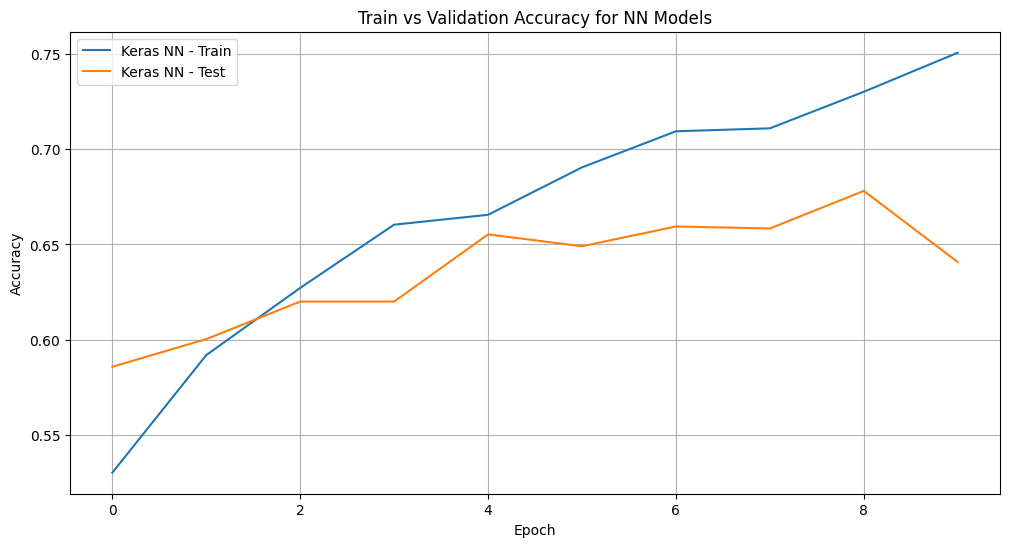
Best BERT Model: Siamese Neural Network with ~71% test accuracy.

## spaCy Results:

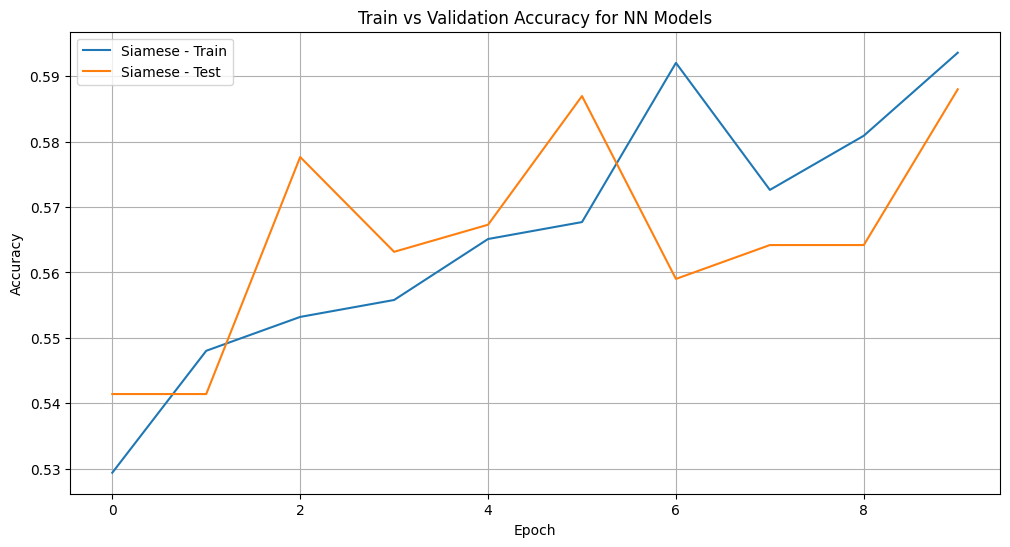
Accuracy Comparison:



Keras NN Accuracy over Epochs:



Siamese Accuracy:



Best spaCy Model: Keras Neural Network with ~64% test accuracy.

**Overfitting Insights**

- Random Forest, XGBoost, and LightGBM showed significant overfitting across all encodings, with nearly perfect training accuracy but large drops on test data.

- Keras and Siamese networks provided better generalization when properly tuned.

**Fusion Model**

To improve performance and robustness, we implemented a Fusion Model using XGBoost as a meta-classifier.

## Inputs to Fusion Model:

- TF-IDF LinearSVC score

- BERT Siamese NN score

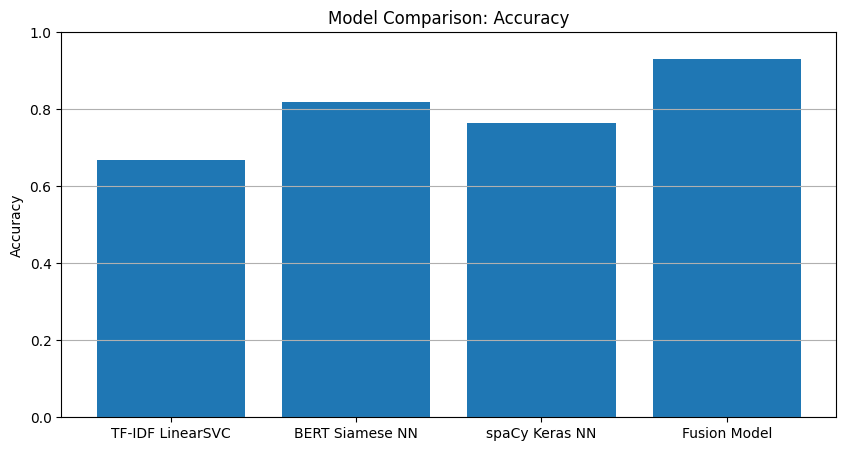
- spaCy Keras NN score

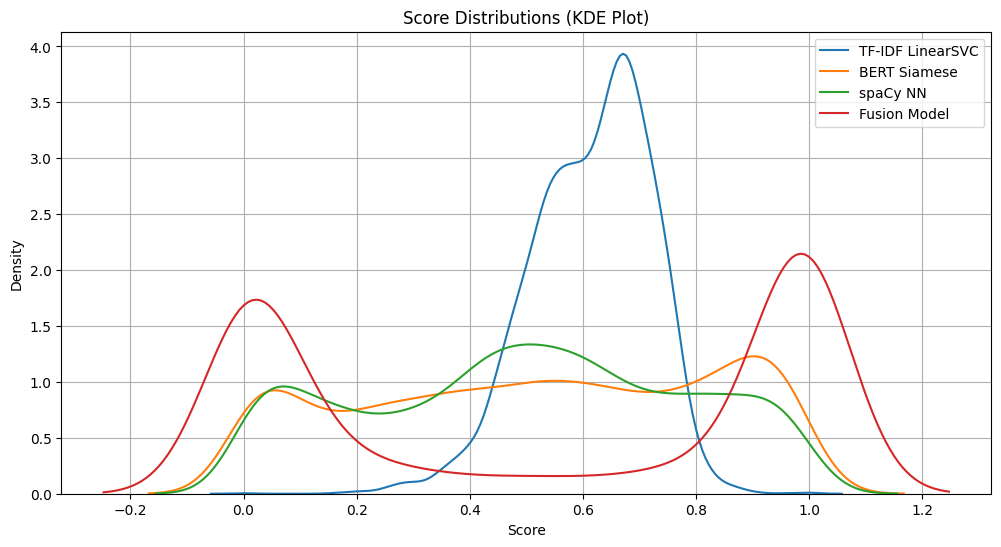
## These were scaled and fed into a final XGBoost model trained to predict acceptance probability. This yielded:

- The most confident score separations (as shown in KDE distributions).

- The most bimodal probability distribution (ideal for ranking tasks).

- Best average accuracy and AUC across the dataset.





## Final ranked CVs can be retrieved using:

get\_ranked\_cvs(job\_id=3)

This returns the top CVs for a specific job, ordered by predicted suitability.  
A black screen with numbers

AI-generated content may be incorrect.