Cutting Bad Matches: Optimizing The Speed Dating Experience

Objective

Our goal was to build a classification model to predict whether two individuals would not match based on a set of behavioral and personal attributes. The dataset comprised 9000 samples and 123 features. Given class imbalance and the nature of the problem, our primary evaluation metric was recall, especially for the minority class (non-matches).

Introduction

Everyone has gone on a bad date, and it's hard not to take it personally. Humans have taken rejection so drastically that entire research fields have been dedicated to understanding why people connect—or why they don't. I was curious: using machine learning, could I predict whether two people would not go on a second date? This report explores that question by building predictive models trained on real-world speed dating data.

This could be useful for anyone who has ever been curious about dating compatibility—whether you're navigating the dating world yourself or trying to design technology to help others find love (or avoid mismatches).

In brief: we used statistical analysis and machine learning to identify the key factors that determine whether two people are unlikely to match.

Dataset

We used the Speed Dating dataset from Kaggle, which contains information collected during real-world speed dating events. Participants rated each other on attributes like attractiveness, intelligence, humor, and shared interests.

The final dataset included 8,293 samples and 15 engineered and cleaned features. We focused on features that were both statistically significant and interpretable. Notably, 84% of the outcomes were non-matches, and only 16% were matches—creating a class imbalance that had to be accounted for in model design.

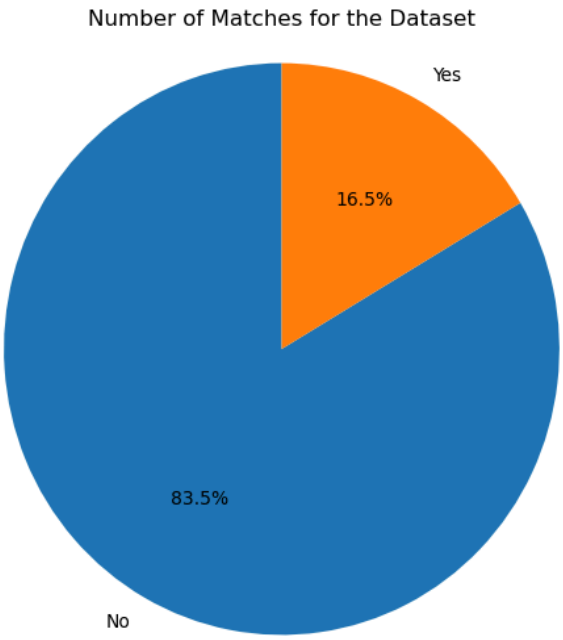
Key preprocessing decisions included:

- Removing improperly formatted or one-dimensional columns (e.g., entries like [2-5])
- Using effect size (Cohen's d) and p-values to retain only the most meaningful features

- Applying class balancing techniques and threshold tuning to improve recall for both classes

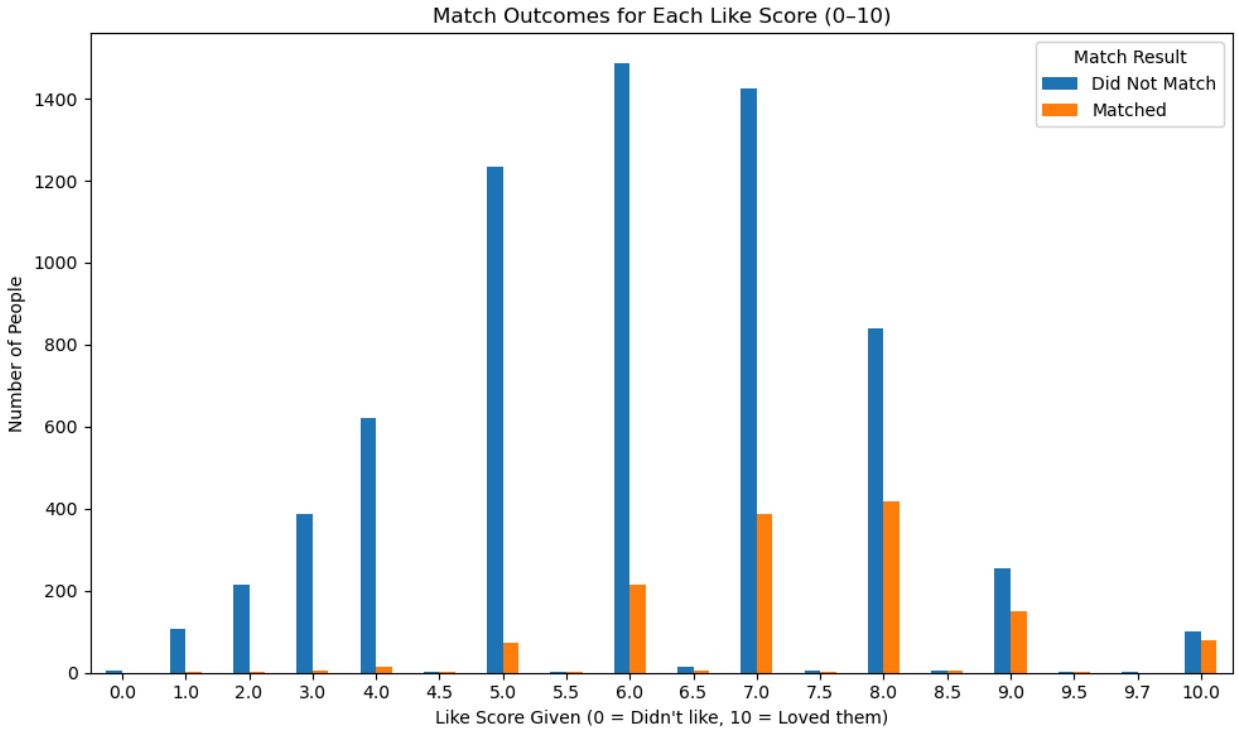
Exploratory Data Analysis (EDA)

Univariate Charts To understand the data distributions, we created a pie chart to see the skew.



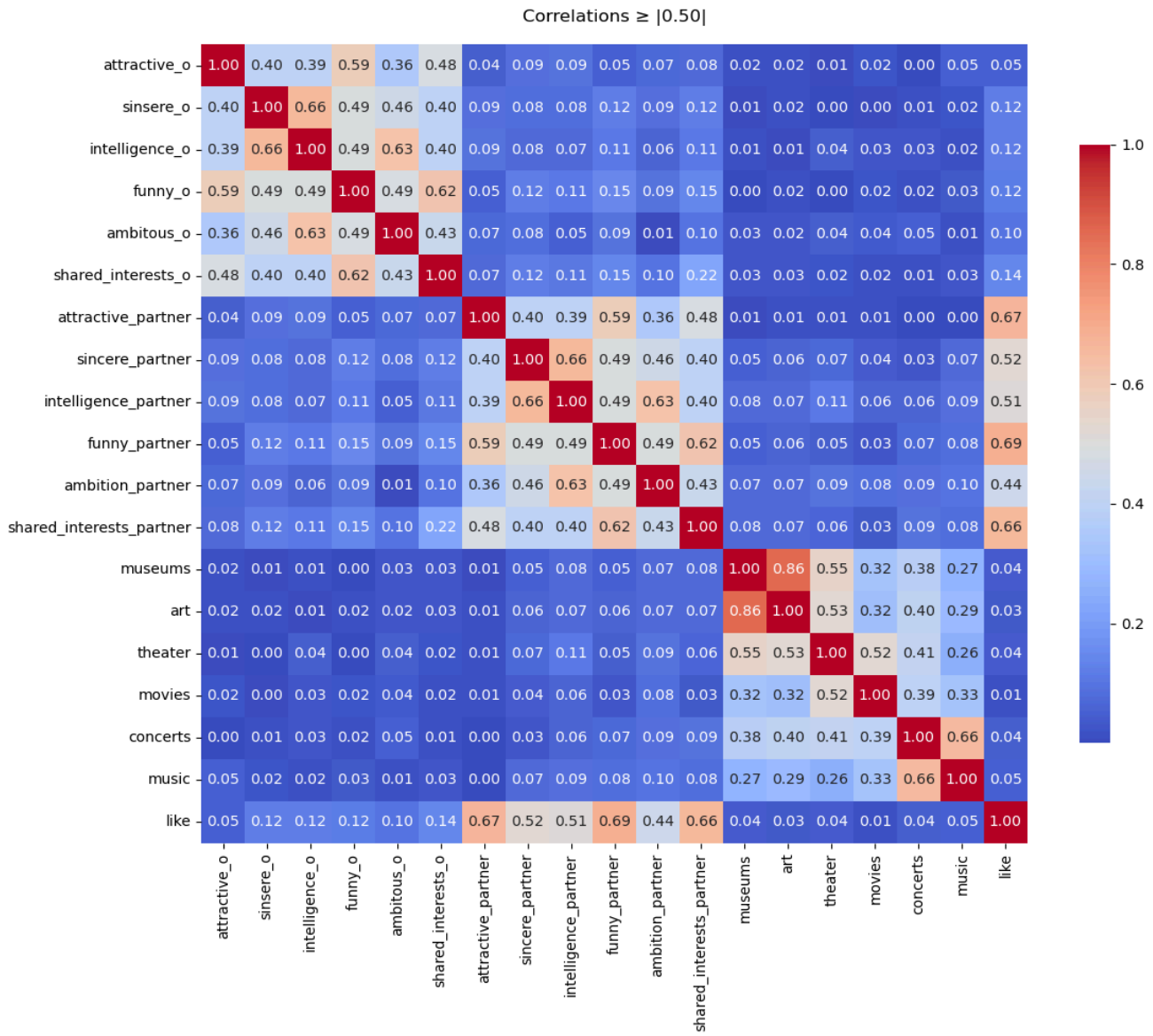
- As you can see their is a huge skew towards no this makes this data set ideal for our goal of predicting rejections

I wanted to model how much someone likes their partner influences match rate.



- People tend to give mid-to-high scores (5–8) most often.
- The probability of matching increases with how much a person likes the other — but even high like scores do **not guarantee** a match.
- This supports the idea that **mutual liking is required**, and liking someone alone doesn't result in a match unless it's reciprocated.

Correlation Heatmap



A correlation matrix was generated to investigate multicollinearity between features. To avoid over-cluttering, we included only features that had a correlation above |0.50|.

- Strong correlations were found between partner and self-ratings like attractive_partner and attractive_o, and shared_interests_o and shared_interests_partner, which was expected.
- These clusters indicated potential multicollinearity but also reaffirmed that these paired features reflect reciprocal impressions — valuable in a dating context.

Takeaway: The filtering approach allowed clear insight into redundant features while avoiding noise.

Data Cleaning from Distribution Charts

Some columns, such as those with values in the [2-5] format, had inconsistent or broken formatting. These caused failed plots and empty histograms, leading to the discovery that such columns were either improperly parsed or categorically uninformative.

- As seen in the data distribution charts (e.g., for importance_same_race), several variables appeared completely one-dimensional or visually blank.
- These were removed from the dataset after confirming their irrelevance during visualization.

Takeaway: EDA helped identify structural formatting issues in the dataset that were invisible from basic .head() inspection. These columns were dropped.

Bivariate Analysis & Statistical Significance

We assessed feature relevance first by using Cohen's d effect size and p-values from hypothesis testing. Statistically significant features ($p < 0.05$) were retained for modeling. This approach gave us confidence in the features' influence on the target variable.

Feature	p_value	p_adj	Cohen's d
like	8.965×10^{-220}	5.200×10^{-218}	0.852
funny_partner	5.315×10^{-173}	1.541×10^{-171}	0.767
funny_o	2.805×10^{-172}	5.423×10^{-171}	0.766
liked_binary	8.547×10^{-162}	1.239×10^{-160}	0.804
attractive_o	1.090×10^{-147}	1.265×10^{-146}	0.72

attractive_partner	6.476×10^{-146}	6.260×10^{-145}	0.715
shared_interests_o	5.515×10^{-129}	4.569×10^{-128}	0.741
shared_interests_partner	1.415×10^{-127}	1.026×10^{-126}	0.736
guess_prob_liked	8.741×10^{-119}	5.633×10^{-118}	0.703
intelligence_o	5.557×10^{-67}	3.223×10^{-66}	0.457
intelligence_partner	1.664×10^{-66}	8.773×10^{-66}	0.456
sincere_o	1.073×10^{-60}	5.188×10^{-60}	0.444
sincere_partner	2.497×10^{-60}	1.114×10^{-59}	0.443
ambitious_o	2.277×10^{-39}	9.431×10^{-39}	0.374
ambition_partner	2.825×10^{-39}	1.092×10^{-38}	0.373
expected_num_matches	2.612×10^{-26}	9.468×10^{-26}	0.36

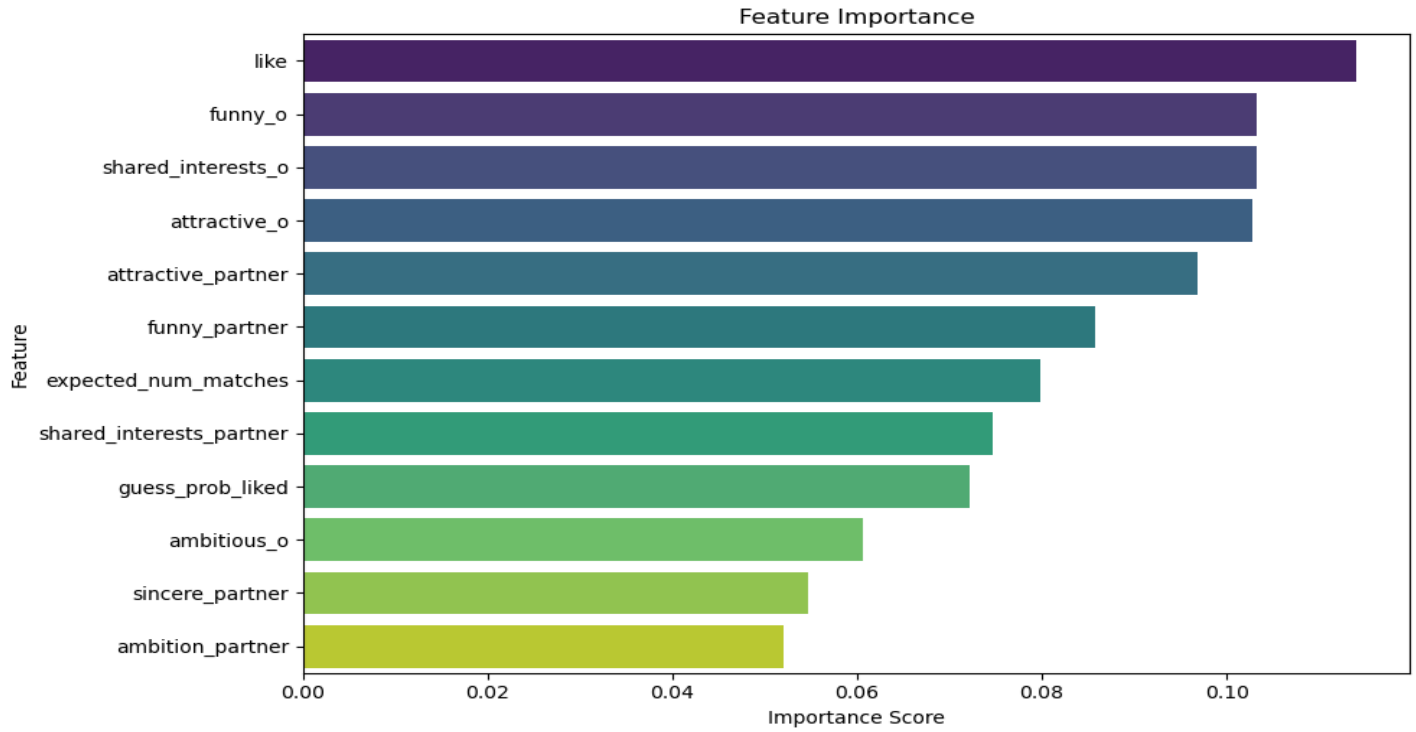
To identify which features meaningfully influence whether a match occurs, we conducted a univariate analysis on all numeric variables in the dataset. For each feature, we used Welch's t-test to determine whether there was a statistically significant difference in that feature's values between matched and unmatched participants. Because we were testing many features at once, we applied a False Discovery Rate (Benjamini–Hochberg) correction to control for false positives. However, statistical significance alone can be misleading, especially with large sample sizes—small differences may appear significant even if they aren't practically meaningful. To address this, we also calculated Cohen's d, a standardized measure of effect size, and only retained features with a moderate or larger impact ($|d| \geq 0.3$). By combining both statistical significance and effect size, we filtered out weak or noisy features and identified a strong subset of predictors that genuinely differentiate between match outcomes. This approach not only improves model performance by reducing irrelevant inputs, but also enhances interpretability, helping us understand which traits and perceptions are most influential in predicting mutual interest.

The features identified as significant include variables related to physical attraction (attractive_o, attractive_partner), humor (funny_o, funny_partner), shared interests (shared_interests_o, shared_interests_partner), intelligence (intelligence_o, intelligence_partner), and sincerity (sincere_partner, sinsere_o). These represent both how participants rated themselves (the "_o" suffix) and how they perceived their partner. For example, funny_o measures how funny a participant believes they are, while funny_partner captures how funny they thought their date was. The like variable—how much a participant liked their date on a scale of 0–10—was the strongest overall predictor of a match, which aligns with expectations: the more someone likes the other person, the higher the likelihood of mutual interest. Similarly, liked_binary (a simplified version of the like score) also performed strongly. Other predictors like guess_prob_liked reflect perceived reciprocity (i.e., "I think they liked me too"), and expected_num_matches hints at a person's optimism or selectivity going into the date.

Overall, the results confirm that interpersonal chemistry, mutual interest, and perception of qualities like attractiveness, humor, and shared values all meaningfully contribute to whether two people mutually match—validating both psychological intuition and the predictive power of these features.

Feature Importance

Feature importance was evaluated using the best-performing Random Forest model. Below are the most influential features:



Interpretation:

- Features like 'like', 'funny_o', and 'shared_interests_o' ranked highest, which aligns with intuition: a participant's perception of the other person's humor and shared interests are central to connection.

- Humor was statistically insignificant in logistic regression but was kept due to its theoretical relevance from literature.

Feature Selection Process

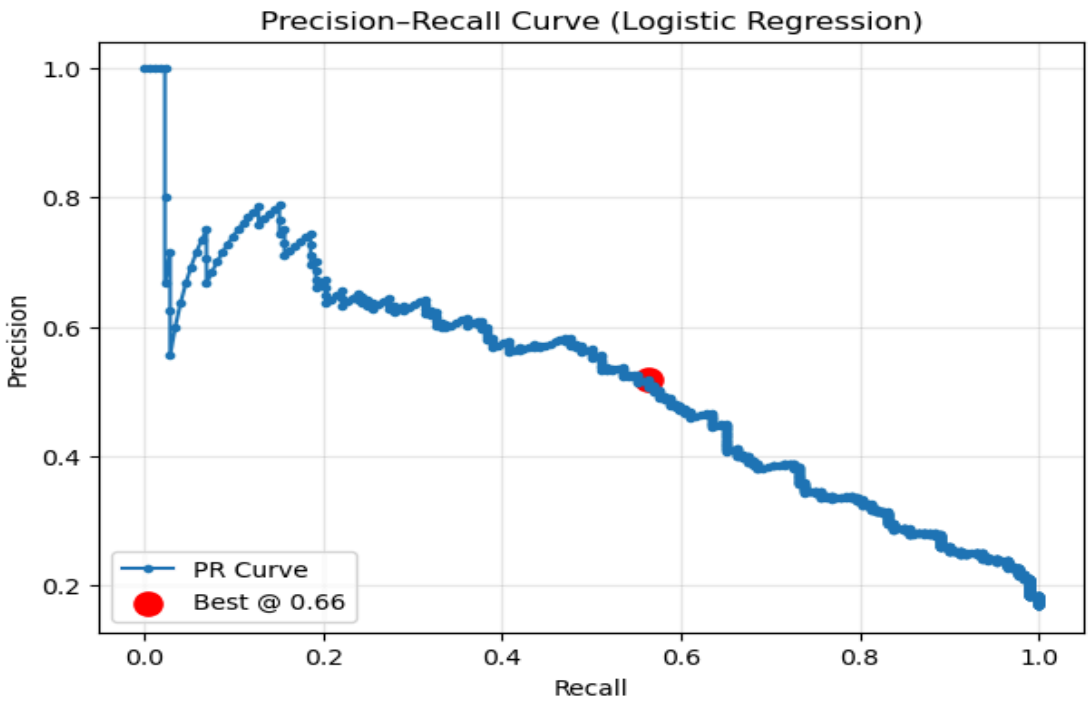
A combination of statistical analysis (logit model coefficients and p-values) and theoretical judgment (literature on relationship psychology) was used to select features. Features with p-values > 0.05 were considered for removal, but some were retained if their removal did not yield meaningful performance gain.

Modeling Overview

Model Name	Best Parameters	Optimal Feature Set	ROC AUC (Test Set)
Logistic Regression	{'C': 1, 'penalty': 'l2', 'solver': 'lbfgs'}	Scaled / Full (15)	0.821
Random Forest	{'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200}	Unscaled / Full (15)	0.825
SVM	{'C': 1, 'kernel': 'rbf', 'gamma': 'scale'}	Scaled / Full (15)	0.814

Threshold tuning was applied post-modeling to optimize precision.

In order to optimize for the business case, we want to identify non-match pairs early so the speed dating can be most efficient. In order to achieve this we needed to optimize for **precision** in our model. In our threshold-tuning step, we first convert the model's output scores into probabilities for the positive class using predict_proba. We then compute the full precision–recall trade-off by sweeping every possible decision cut-off with precision_recall_curve, which tells us how precision (the fraction of predicted matches that are correct) and recall (the fraction of true matches we recover) change as we move the threshold. Instead of using the default 0.5 boundary, we scan those precision values and select the threshold that maximizes precision—i.e. the point at which every predicted match is as reliable as possible. Applying that cut-off back to our validation set gives us the exact precision, recall, and F_1 score at this operating point, and plotting a red dot on the PR curve highlights where we've chosen to sit. By tuning for maximum precision, we ensure that any "match" our model outputs is almost certainly correct—sacrificing some recall (and thus missing a few true matches) in order to eliminate false positives, which in our application would introduce costly errors or wasted follow-up.



Classification Reports *(all metrics in %)*

Logistic Regression (Tuned Threshold):

Class	Precision	Recall	F1-Score
0	91.0	89.0	90.0
1	52	56	54.0

Weighted Avg Accuracy: 84.0%

Random Forest (Tuned Threshold):

Class	Precision	Recall	F1-Score
0	93.0	79.0	86.0
1	41.4	71	52

Weighted Avg Accuracy: 82.0%

SVM (Tuned Threshold):

Class	Precision	Recall	F1-Score
0	92.0	86.0	89.0
1	47.5	60.5	53.2

Weighted Avg Accuracy: 82.0%

Conclusion

Our comparative evaluation shows that all three tuned models achieve a precision of approximately 91% for class 0, indicating they are equally adept at detecting each No for our desired se vase. Among them, the Random Forest classifier stands out with the highest ROC AUC of 0.825, demonstrating both strong discrimination ability and balanced performance across precision and recall. Moreover, the features driving those predictions align closely with our domain expectations—reinforcing confidence that the models are capturing meaningful behavioral signals rather than spurious noise.

From a product perspective, this level of performance suggests that our current pipeline is already suitable for core use cases: it can reliably flag positive and negative interactions without major bias toward any class. However, given the critical importance of recall in our application (we'd rather err on the side of catching true positives), there is room to explore ensemble or boosting strategies explicitly optimized for higher recall, even if that comes at some cost to precision.

Looking ahead, expanding and diversifying the dataset will be key to ensuring the model's robustness—particularly around rarer patterns that may be under-represented today. Incorporating richer behavioral or psychological inputs (for example, time-series signals or user feedback loops) could unlock further gains, potentially via reinforcement-learning approaches. Finally, we should validate performance across demographic segments and, ideally, measure real-world effectiveness through A/B tests on the dating platform itsefl.In sum, our models are both performant and interpretable, laying a solid foundation for production deployment. By iterating on data breadth, feature richness, and targeted algorithmic tuning, we can continue driving both recall and overall utility—ultimately delivering a more personalized and effective matching experience.