

Predictive Modeling on Real Hiring Data: Exploring Salary Stereotypes and Bias Mitigation

By Lyudmil Stamenov

Data Overview & Key Challenges

Dataset Overview

- **Total Columns:** 39
- **Includes:**
 - Candidate information
 - Job position details
 - Current interview stage
 - Final hiring outcome
- **Source:** Akkodis

Data Challenges

- **Duplicate Candidate IDs**
 - Result of merging data from two distinct databases
- **Missing Position Data for Rejected Candidates**
 - Job position details are only available for hired candidates
 - Limits the ability to learn from negative outcomes (i.e., rejections)
- **Multiple Rows per Application**
 - Each application includes one row per interview step
- **Inconsistent City Names**
 - Translation from Italian to English introduced naming errors and mismatches
- **Unstandardized Text Fields**
 - Columns like job titles and study areas require semantic matching
- **Strong Class Imbalance**
 - Unequal distribution of hired vs. rejected candidates affects model training
- **Many NaN Values**
 - Missing data in key columns requiring careful imputation or exclusion

Made with GAMMA

Preprocessing Steps

Deduplication of Candidate IDs

Split IDs using stable candidate attributes, including:

- Sex, Job Title Hiring
- Study Area, Study Title
- Assumption Headquarters, Residence
- Year of insertion, Age Range
- Years of Experience

Filtering Candidates with Inconsistent Outcomes

Excluded candidates who:

- Withdrew early due to personal reasons
- Were approved but not hired because of factors like competing candidates, lost availability, pending departures, closed opportunities, or voluntary withdrawal

Aggregating Interview Steps

Consolidated multiple interview stage rows into a single data point per application

Generating Synthetic Negative Samples

- For positively labeled candidates: paired with a random position of low similarity
- For negatively labeled candidates: paired with two random positions, one with high similarity and one with low similarity

Residence Data Standardization

- Created a custom mapping to correct or approximate city names using actual or nearby locations
- Calculated distance (km) between candidate residence and Akkodis Headquarters as well as Assumption Headquarters
- Added boolean flags for Italian Residence and European Residence

Text Matching Techniques

Evaluated various text embedding models to calculate similarity scores across job-related fields

Addressing Class Imbalance

Tested multiple sampling techniques (SMOTE variants, ADASYN, etc.) to balance classes effectively

Handling Missing Values

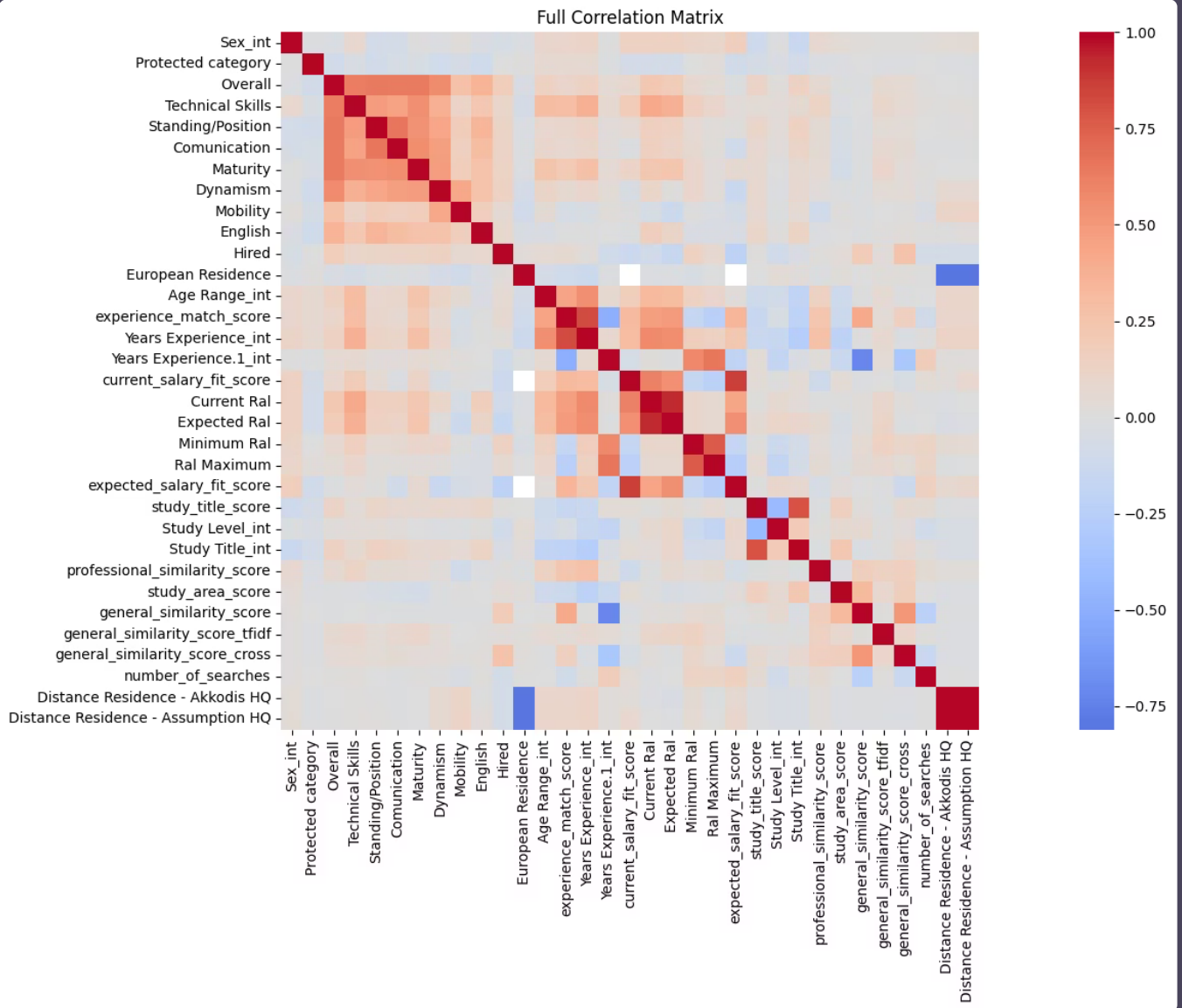
- Used ML models capable of handling NaNs directly
- Applied Simple Imputer with mean strategy for other models

Feature Engineering

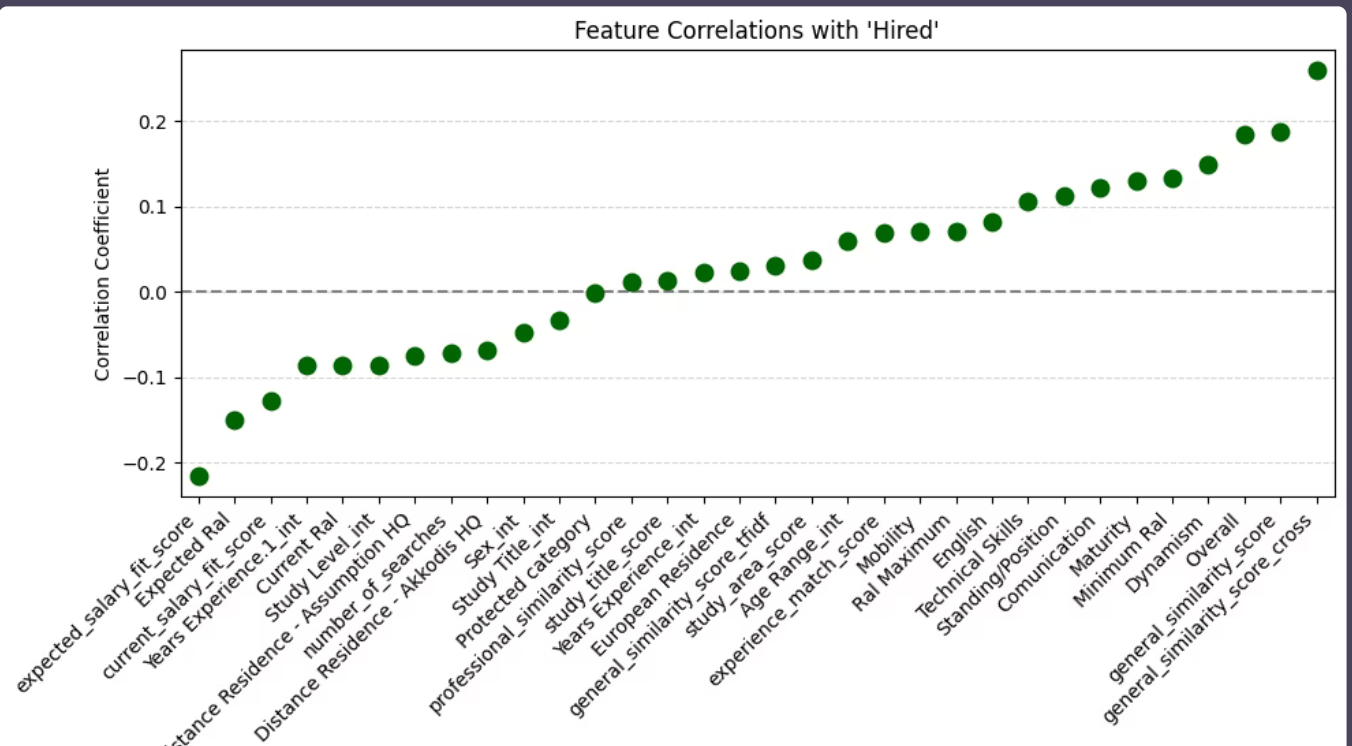
- **Distance to Assumption HQ**
Calculated using the geographic coordinates of the candidate's residence and the Assumption Headquarters.
- **Distance to Akkodis HQ**
Calculated using the geographic coordinates of the candidate's residence and the Akkodis Headquarters.
- **Candidate Text Profile**
A combined textual field aggregating:
`Study Title`, `Study Area`, `Sector`, `Last Role`, `Years of Experience`, and `Tags` (skills noted by the recruiter).
- **Job Text Profile**
A combined textual field aggregating:
`Job Title`, `Job Family`, `Recruitment Request`, `Job Description`, `Candidate Profile`, `Required Study Level`, `Required Study Area`, and `Required Years of Experience`.
- **General Similarity Score**
Cosine similarity between candidate and job text using **sentence encoder embeddings**.
- **Cross-Encoder General Similarity Score**
Cosine similarity computed with a **cross-encoder model** between candidate and job text.
- **TF-IDF General Similarity Score**
Cosine similarity using **TF-IDF vectors** of the candidate and job text.
- **Experience Match Score**
Normalized categorical difference between `Years of Experience` and `Required Years of Experience`.
- **Current Salary Fit Score**
Assesses if `Current RAL` (salary) falls within the range `[Min RAL, Max RAL]`; if not, computes the normalized distance from the nearest bound.
- **Expected Salary Fit Score**
Assesses if `Expected RAL` (salary) falls within the range `[Min RAL, Max RAL]`; if not, computes the normalized distance from the nearest bound.
- **Study Title Match Score**
Normalized difference between `Study Title` and required `Study Level`.
- **Professional Similarity Score**
Cosine similarity of encoder embeddings between concatenated candidate fields (`Sector + Last Role`) and job fields (`Job Family + Job Title`).
- **Study Area Similarity Score**
Cosine similarity between encoder embeddings of `Study Area` and required `Study Area`.

Dataset Analysis

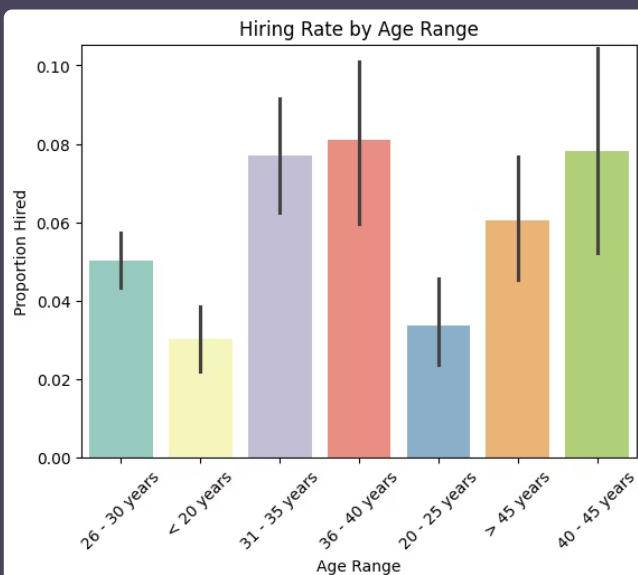
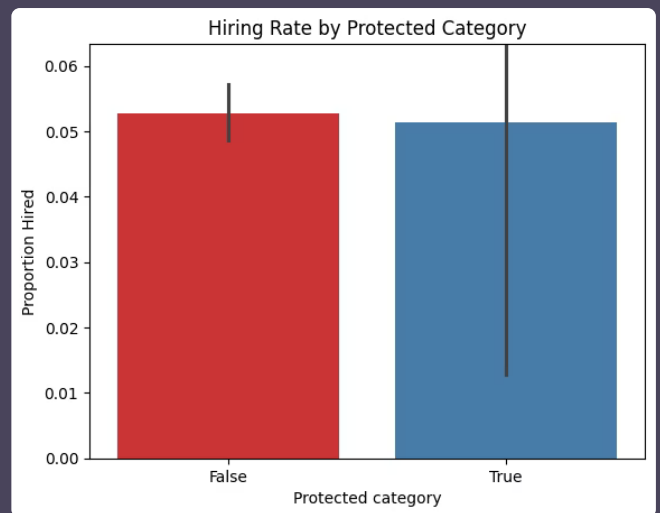
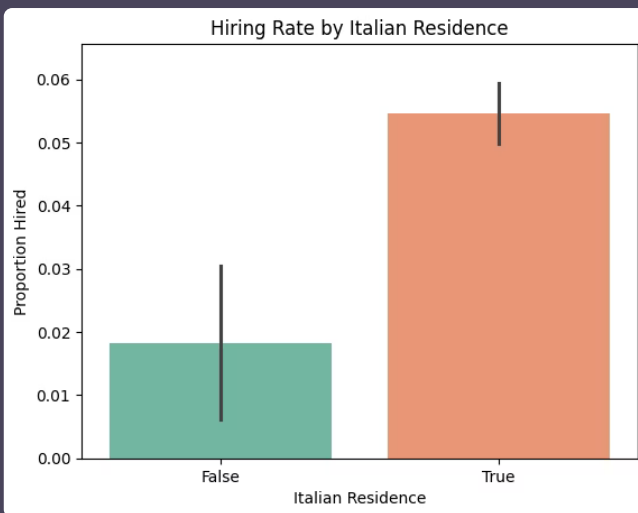
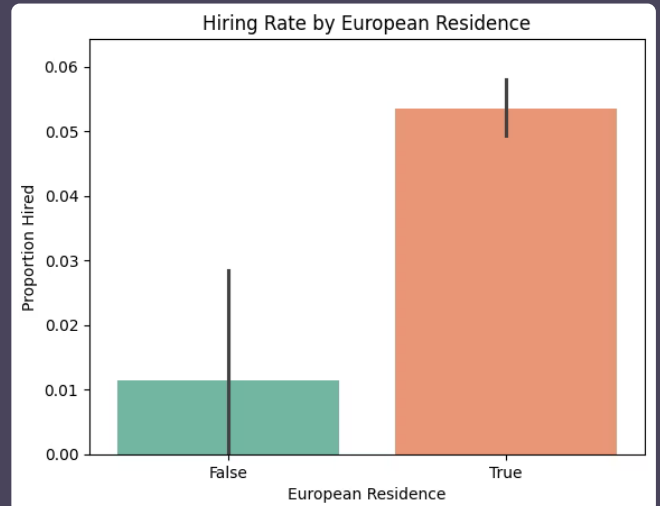
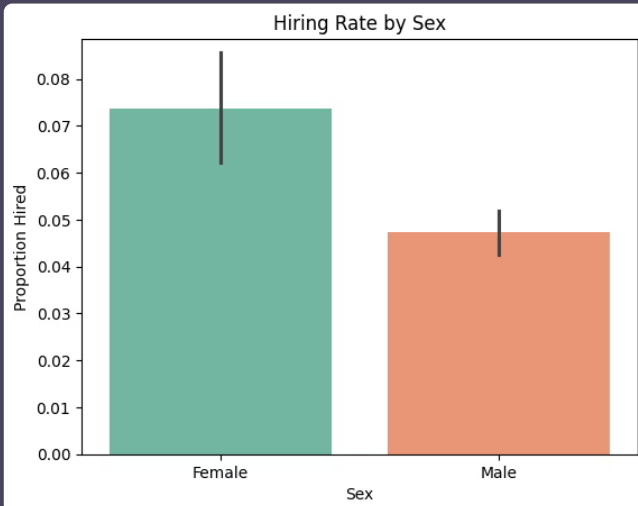
Correlation Matrix



Features' Correlations with "Hired"



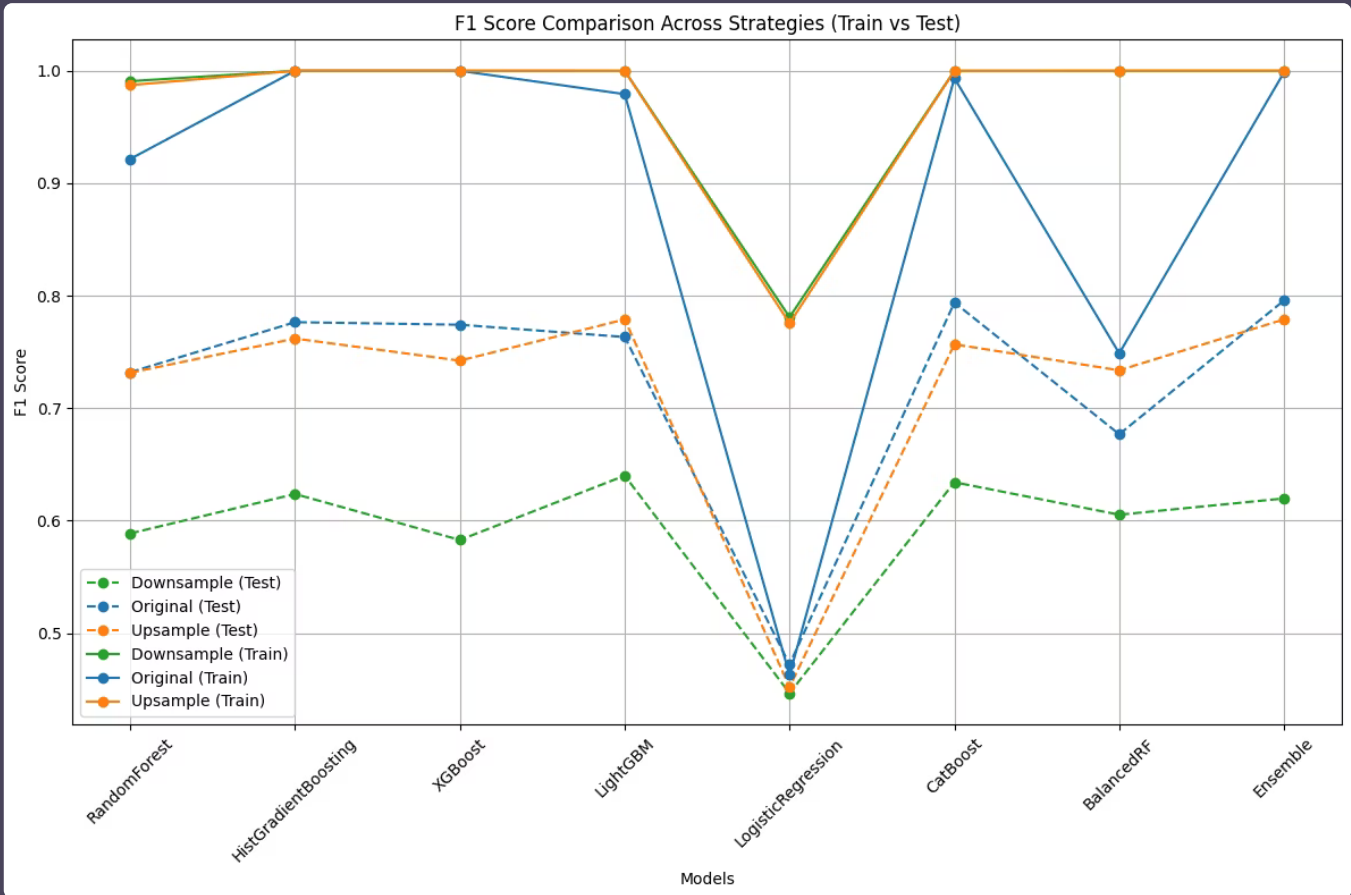
Hiring rate by protected column



Models Comparison

Model	Original Sampling	Downsample Sampling	Upsample Sampling
Ensemble	0.795812	0.619718	0.778947
CatBoost	0.793970	0.634146	0.756757
HistGradientBoosting	0.776471	0.623656	0.761905
XGBoost	0.774194	0.582781	0.742268
LightGBM	0.763285	0.640000	0.778947
RandomForest	0.731959	0.588608	0.731481
BalancedRF	0.676806	0.605263	0.733668
LogisticRegression	0.472131	0.446541	0.452012

The table compares F1 scores across different models and sampling methods. Original sampling showed the best performance for most models.



Similarity Computation



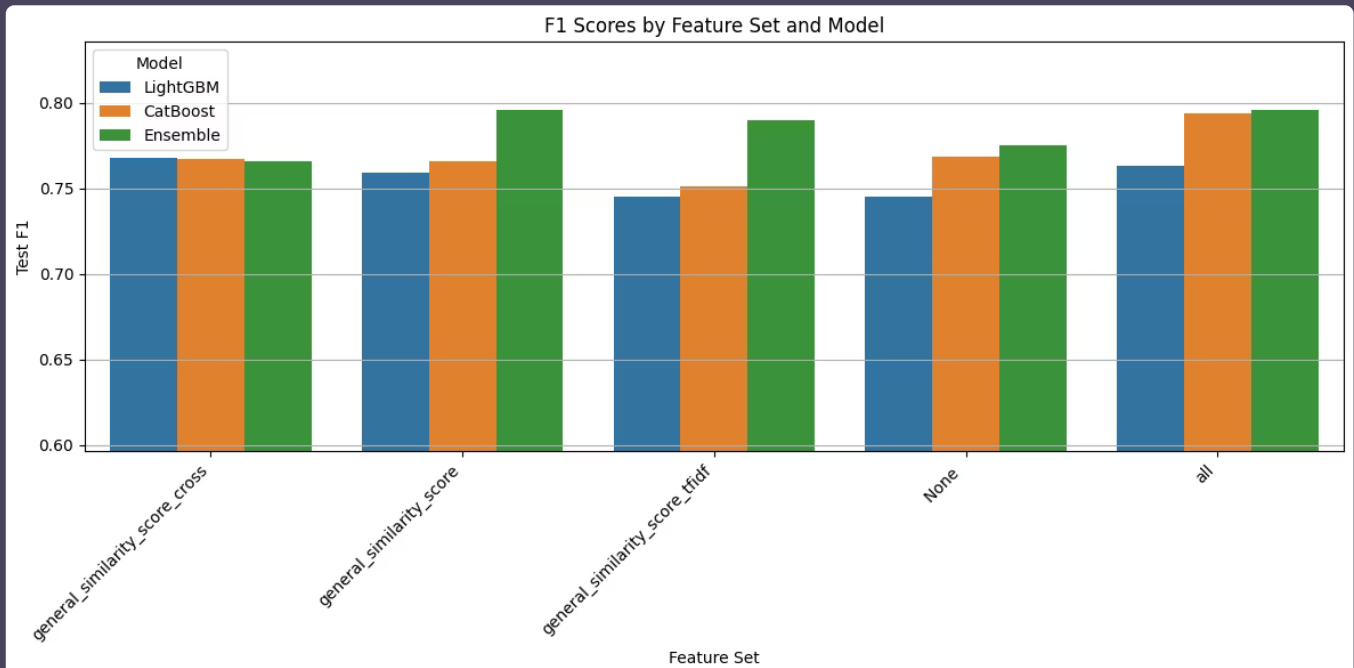
TF-IDF (Term Frequency-Inverse Document Frequency)



Sentence Encoders



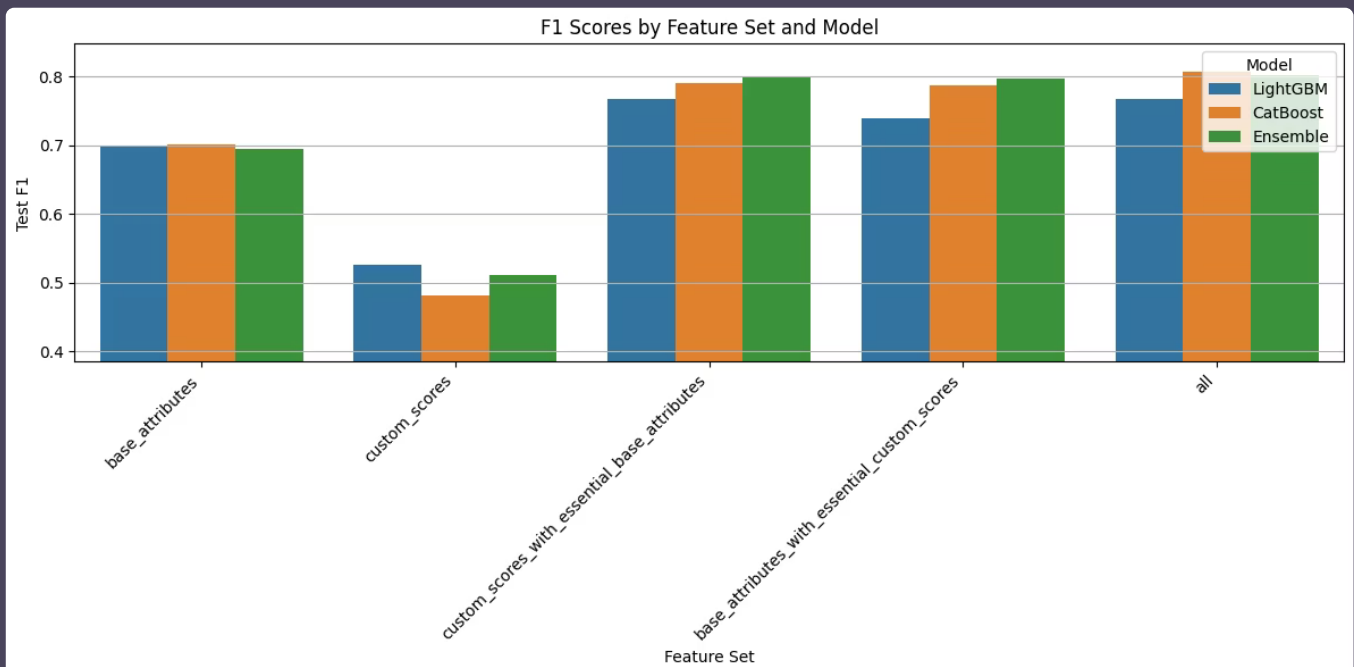
Cross Encoder



Feature Set Selection

Feature Set Options:

- **Base attributes:** Sex, Protected Category, Overall, Technical Skills, Position, Communication, Maturity, Age, Italian Residence, Years Experience, etc.
- **Custom scores:** Similarity scores computed between candidate and job text fields, such as experience_match_score, current_salary_fit_score, expected_salary_fit_score, study_title_score, professional_similarity_score, study_area_score, general_similarity_score, and distances to Akkodis HQ and Assumption HQ.
- **Custom scores + essential base:** Combines custom scores with base attributes not involved in score calculations (e.g., Sex, Protected Category).
- **Base attributes + essential custom scores:** Base attributes plus essential custom scores derived from non-categorical fields whose original (predecessor) features cannot be used alone in the model, including study_area_score, general_similarity_score, and distances to HQs.
- **All attributes:** Full set of base and custom features.



Sampling Techniques

Sampling Techniques & Models Tested

Techniques Tried:

- Original dataset (no resampling)
- SMOTE at 25%, 50%, 75%, and 100% oversampling
- ADASYN at 25%, 50%, 75%, and 100% oversampling
- Borderline-SMOTE at 25%, 50%, 75%, and 100% oversampling
- SVM-SMOTE at 25%, 50%, 75%, and 100% oversampling
- Downsampling

Models Tested:

- LightGBM
- CatBoost
- Ensemble

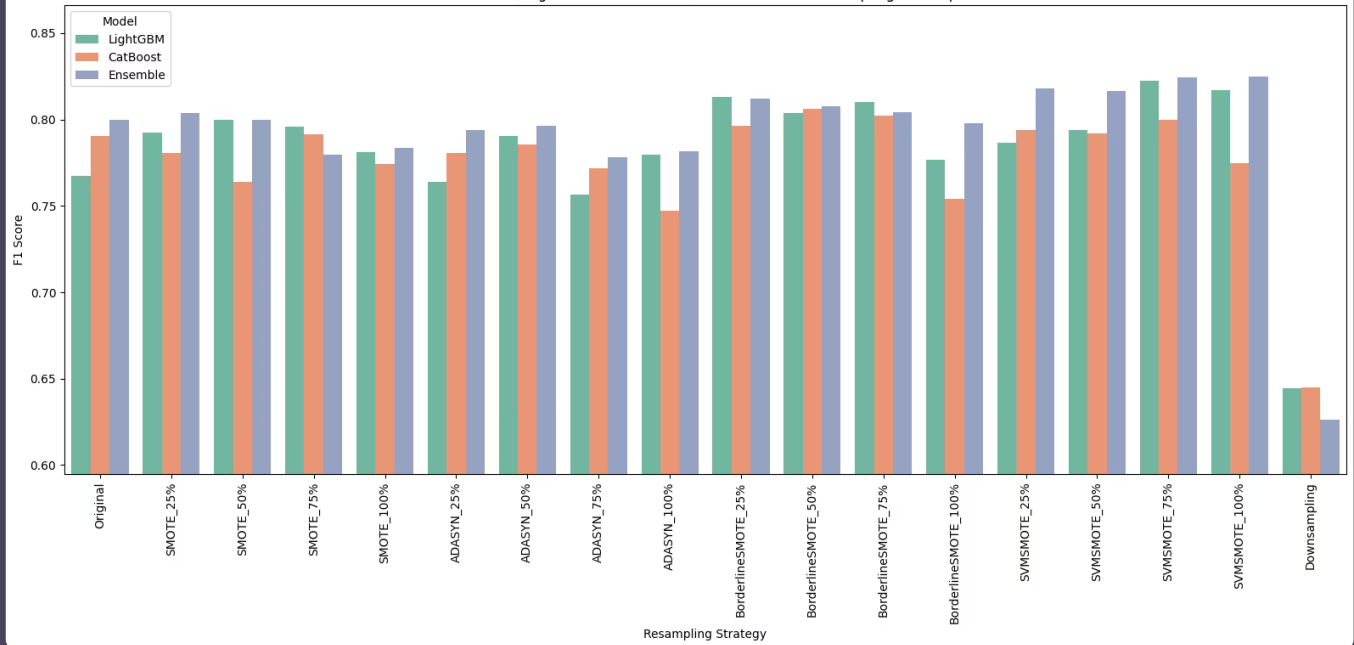
Performance Summary (Train F1 / Test F1):

Sampling Method	Model	Train F1	Test F1
Original	LightGBM	0.93	0.77
Original	CatBoost	0.99	0.79
Original	Ensemble	1.00	0.80
SMOTE 50%	LightGBM	0.99	0.80
BorderlineSMOTE 25%	LightGBM	0.98	0.81
SVM SMOTE 75%	Ensemble	1.00	0.82
Downsampling	Ensemble	1.00	0.63

Insights:

- SMOTE, BorderlineSMOTE, and SVM SMOTE variants boost test performance, especially with LightGBM and Ensembles.
- Downsampling shows poor generalization despite perfect training scores.
- Ensembles consistently outperform individual models on test data.

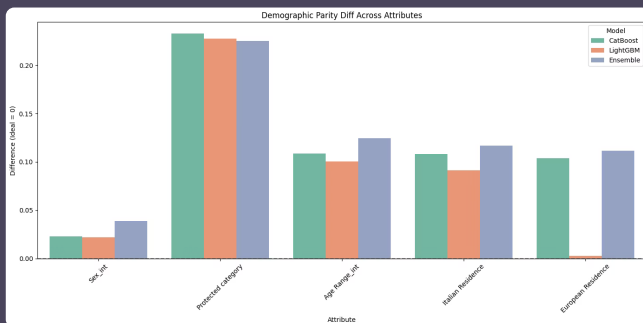
F1 Score of CatBoost, LightGBM and Ensemble Models Across Resampling Techniques & Ratios



Fairness Metrics Utilized

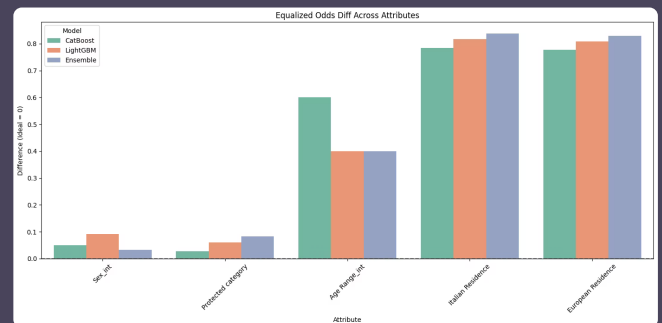
Demographic Parity Difference

Measures the difference in the proportion of positive outcomes between different demographic groups, aiming for equal representation in results.

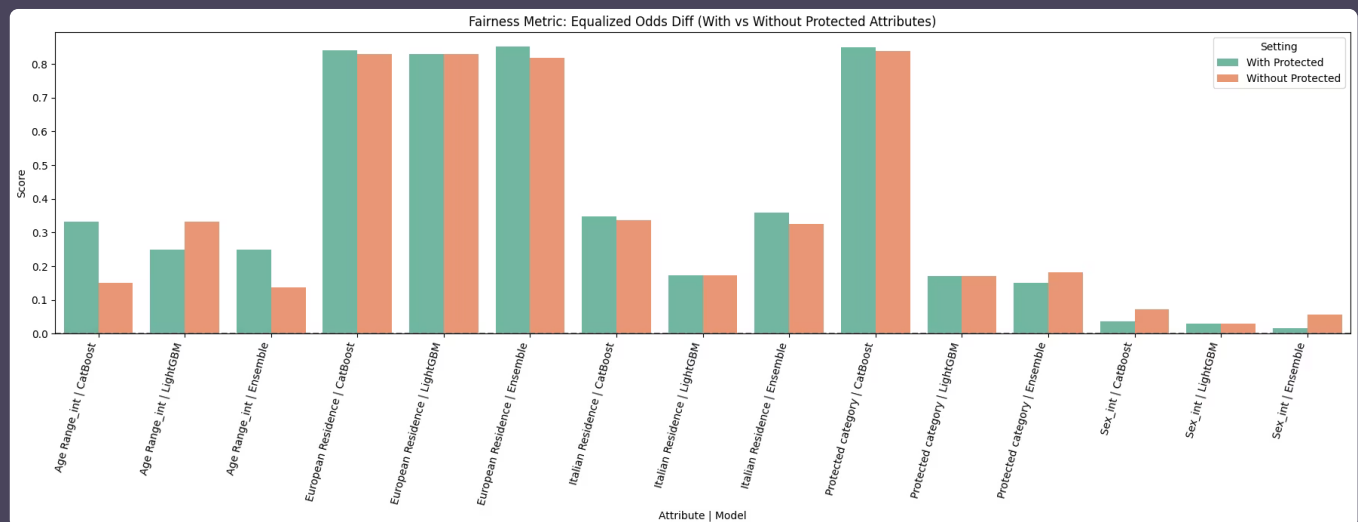
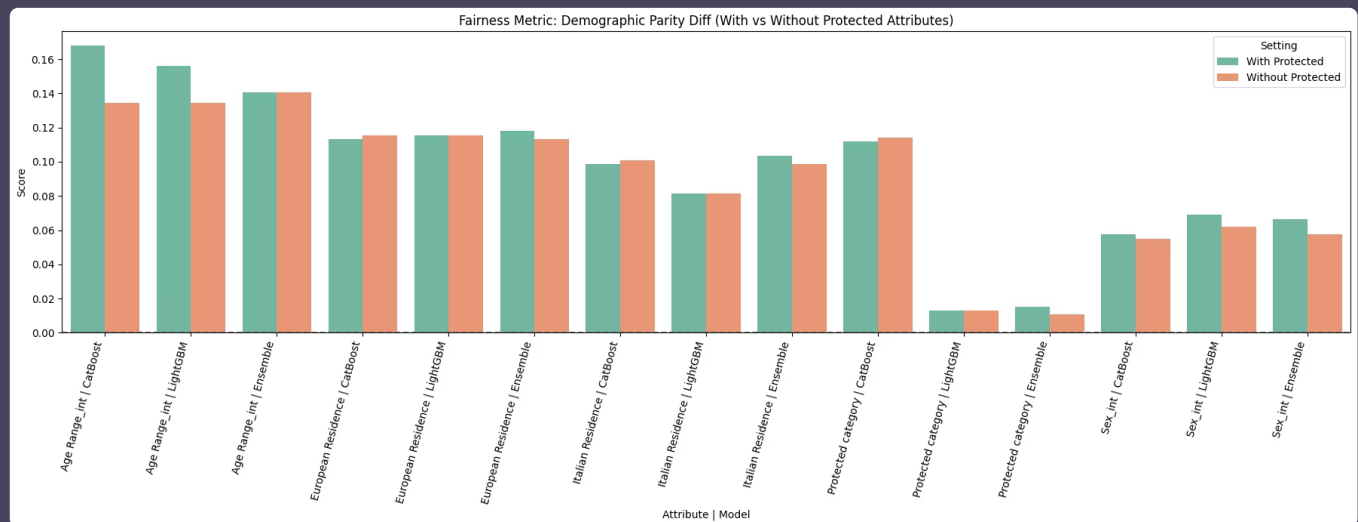


Equalized Odds Difference

Evaluates fairness by ensuring that protected groups have equal true positive rates and equal false positive rates, addressing both prediction accuracy and error types.



Dropping Protected Attributes



Bias Mitigation Techniques

Preprocessing

- **Reweighting:**
Adjusts weights of samples to balance representation across sensitive groups before training.
- **Disparate Impact Remover:**
Modifies feature values to reduce bias while preserving data utility.

In-processing

- **FAUCI:**
Incorporates fairness constraints directly into the model training process to reduce bias.
- **Prejudice Remover:**
Adds a regularization term to penalize biased predictions during learning.

Conclusion

- Tested bias mitigation methods showed mixed effectiveness across sensitive attributes.
- Preprocessing techniques (Reweighting, Disparate Impact Remover) maintained model accuracy but had limited bias reduction.
- In-processing methods (Fauci, Prejudice Remover) reduced some fairness disparities but sometimes hurt model stability or gave inconsistent results.
- No single method fully resolves bias while preserving performance.
- A combined or hybrid approach is likely needed to balance fairness and predictive quality effectively.

Conclusion & Future Work

Conclusion

- Tackled multiple data challenges (duplication, missing values, unstructured text, class imbalance) with a comprehensive preprocessing pipeline.
- Demonstrated that combining **feature selection**, **resampling**, and **custom similarity scores** significantly improves model performance.
- Applied several **bias mitigation techniques** (Reweighting, Disparate Impact Remover, Prejudice Remover, Fauci Method), but fairness improvements remained **limited and inconsistent**, especially for geographic attributes.
- Results show that **improving accuracy and fairness simultaneously remains a challenge**.

Future Work

- Explore **more advanced bias mitigation models** and hybrid techniques.
- Focus on improving fairness metrics without compromising model utility.
- Refine **semantic matching** techniques for text-based fields using newer embeddings or large language models.