ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO DI

INFORMATICA – SCIENZA E INGEGNERIA – DISI

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

TESI DI LAUREA

in

INTELLIGENT SYSTEMS

Fairness in Human Resource — Bias Analysis and Mitigation

RELATORE

Prof. Andrea Borghesi

CORRELATRICE

Prof.ssa Roberta Calegari

CANDIDATA

Giulia Vivarelli

IV 03/2025

Anno Accademico 2023/2024

Sommario

1. Introduzione ***4***

2. AI Fairness ***6***

2.1. Il concetto di Fairness **6**

2.2. Sorgenti Unfairness **8**

2.3. Bias Detection **9**

2.4. Bias Mitigation **12**

2.4.1. Pre-processing **13**

2.4.2. In-processing **14**

2.4.3. Post-processing **14**

2.5. Limiti **15**

3. Dataset Akkodis ***17***

3.1. Struttura del Dataset **17**

3.2. Preprocessing **22**

3.2.1. Overview **22**

3.2.2. Data Cleaning **22**

3.2.3. Feature Mapping **23**

3.2.4. ID duplicati **25**

3.2.5. Dataset Sorting **26**

3.3. Scelta del Target **27**

3.3.1. Rimozione Candidati nelle prime fasi di recruiting **28**

3.3.2. Rimozione Candidati validi **29**

3.4. Creazione dataset finale **30**

3.4.1. Colonne Accessorie **32**

4. Addestramento ***34***

4.1. Preprocessing **34**

4.2. Analisi del dataset **35**

4.2.1. Matrice di correlazione **35**

4.2.2. Colonna Target **37**

4.2.3. Analisi Attributi Sensibili **37**

4.3. Scelta dei modelli **44**

4.3.1. Rete Neurale **44**

4.4. Addestramento **46**

4.4.1. Modelli Tradizionali **46**

4.4.2. Rete Neurale **48**

5. Analisi dei Bias ***52***

5.1. Metriche scelte 52
5.2. Implementazione Metriche 52
5.2.1. Demographic Parity 53
5.2.2. Equalized Odds 53
5.2.3. Counterfactual Fairness 54
5.3. Bias Detection 55
5.3.1. Equalized Odds 57
5.3.2. Counterfactual Fairness 57
5.4. Explainability 58
5.4.1. LIME 58
5.4.2. SHAP 62
6. Mitigazione dei Bias 64
6.1. Mitigazione pre-processing 64
6.1.1. Rimozione Attributi Biased 64
6.1.2. Data Augmentation 65
6.1.3. Bilanciamento Dataset 68
6.2. Mitigazione in-processing: Adversarial Debiasing 71
6.2.1. Architettura 71
6.2.2. Equalized Odds Penalty 72
6.2.3. Addestramento 72
6.2.4. Selezione Soglia Ottimale 74
6.3. Mitigazione post-processing: Fair Thresholding 76
7. Discussione 78
Bibliografia 80

1. Introduzione

Negli ultimi anni l'intelligenza artificiale è diventata parte integrante della vita di tutti i giorni, offrendo la possibilità di automatizzare e velocizzare processi altrimenti complessi. Il suo sviluppo ha però sollevato preoccupazioni riguardanti la potenziale discriminazione nei confronti di categorie protette, soprattutto in contesti sensibili come quello sanitario, del *recruiting* o assicurativo.

Sebbene i pregiudizi possano sembrare puramente umani, è noto come la maggior parte dei modelli di *machine learning* riflettano e addirittura amplifichino i *bias* dei contesti in cui sono sviluppati.

Per questo motivo il concetto di *fairness* ha acquisito un ruolo centrale anche in ambito tecnologico, nel quale è fondamentale garantire che gli algoritmi non perpetuino le disuguaglianze già ampiamente radicate nella società.

Esistono due principali sorgenti di *unfairness*: dataset *biased* e algoritmi *biased*, frutto di scelte di design disinformate. Questi ultimi possono generare un effetto a catena: algoritmi *biased* generano nuovi dati *biased* che influenzano negativamente le decisioni umane e che possono essere utilizzati a loro volta per l'addestramento di altri modelli.

Per contrastare gli effetti dei dataset *biased*, la strategia più logica sarebbe quella di utilizzare dataset "neutrali", che non contengano disuguaglianze. Tuttavia, dataset di dimensioni adeguate all'addestramento di modelli di AI sono normalmente il risultato di anni di evoluzione sociale e culturale, con tutte le sfumature che ne derivano. In aggiunta, la maggior parte delle categorie protette tende ad essere sottorappresentata, rendendo ancora più complessa la ricerca di dataset bilanciati.

A questo scopo ricercatori e aziende da tutto il mondo hanno sviluppato e continuano a sviluppare metriche per l'identificazione di queste anomalie e tecniche per mitigarne gli effetti.

Questa tesi affronta il tema della *fairness* nell'ambito del recruiting, utilizzando dati reali, forniti dall'azienda **Akkodis**. Il dataset viene utilizzato per addestrare modelli di machine learning a riconoscere candidati idonei per una data posizione lavorativa, sulla base dei dati forniti, come titolo di studio, settore, anni di esperienza.

L'obiettivo è analizzare se l'impiego dell'AI in questo ambito, oltre che a velocizzare il processo di *recruiting* automatizzandone le prime fasi, possa introdurre discriminazione nei confronti di categorie protette. Insieme all'analisi delle prestazioni di ciascun modello viene valutato, attraverso metriche selezionate, il livello di equità preliminare.

I risultati ottenuti sono analizzati tenendo conto del contesto in cui sono stati raccolti. Akkodis è infatti una azienda che si impegna attivamente per l'inclusività, e questo si riflette sui suoi dati, che evidenziano come le scelte di assunzione siano giustificate dalle competenze di ciascun candidato.

Questo progetto include il preprocessing del dataset, importato da un file in formato Excel, l'addestramento di diverse tipologie di modelli, la valutazione del livello di equità e l'applicazione di diverse tecniche di mitigazione.

2. AI Fairness

2.1. Il concetto di Fairness

Come spesso accade con le grandi innovazioni tecnologiche, l'AI è diventata parte integrante della società ed ha acquisito un ruolo importante anche in settori critici, come la sanità, il lavoro e la giustizia. La sua diffusione in questi ambiti è stata spinta dalla promessa di rendere i processi più veloci, efficienti e possibilmente più oggettivi.

Sebbene i pregiudizi siano spesso associati alla natura umana, contrapposta, nell'immaginario comune, alla neutralità fredda di un algoritmo, i fatti espongono una realtà ben diversa. Gli algoritmi di *machine learning* sono infatti addestrati per emulare il comportamento umano e sono per questo soggetti anch'essi a *bias*.

Per questo motivo l'equità ha acquisito importanza anche in ambito tecnologico, nel quale sarebbe necessaria una definizione rigorosa e universale. Nell'ambiente dell'AI si fa riferimento all'equità come **AI Fairness.**

Tuttavia, si tratta di un concetto astratto e soggettivo, strettamente legato al contesto, ai valori sociali degli utilizzatori e agli obiettivi specifici di un sistema. In particolare, negli algoritmi di *machine learning* la *fairness* entra spesso in conflitto con le prestazioni richieste, rendendo necessario un compromesso tra accuratezza e imparzialità. Chi sviluppa questi sistemi si trova a dover bilanciare le richieste degli *stakeholder*, con i requisiti imposti dalle nuove leggi a riguardo.

Nondimeno, l'imparzialità viene spesso messa in secondo piano per dare priorità alle *performance*. Spesso il target di queste applicazioni non è l'intera comunità ma un sottogruppo ristretto di utilizzatori, che nella maggior parte dei casi non rientra tra le categorie protette ed è maggiormente rappresentato nei dataset utilizzati per l'addestramento.[1]

Nell'ambito dell'intelligenza artificiale si definiscono come *fair* modelli imparziali, che non producono risultati sfavorevoli sulla base di attributi sensibili, come sesso, età, etnia. L'impatto di un modello *unfair* può essere significativo, soprattutto in contesti critici.

[1] Madaio et al., «Assessing the Fairness of AI Systems».

Un esempio concreto è dato dall'algoritmo di raccomandazioni lavorative di *Google*, che tendeva a raccomandare posizioni lavorative più redditizie a candidati di sesso maschile. Un caso simile è emerso poi con l'algoritmo di selezione sviluppato da *Amazon*, ritirato a causa della sua tendenza nell'associare punteggi inferiori a candidati di sesso femminile. In questo caso la disparità è stata attribuita alla sottorappresentazione delle donne all'interno del dataset di addestramento.[2]

La discriminazione nei sistemi di AI può essere classificata in:

- **Discriminazione spiegabile**, attraverso la trasparenza sull'utilizzo di attributi, non sensibili, che la giustifichino. Per esempio, analizzando il reddito annuo, potrebbe emergere che le donne hanno un guadagno più basso rispetto agli uomini. Tuttavia, questo potrebbe essere giustificato dal numero di ore lavorative, che possono cambiare tra candidati di sesso diverso e sono alcune volte inferiori per le donne. Nel caso di ore lavorative inferiori per le donne la discriminazione si dice legale e cercare di bilanciarla porterebbe a discriminazione inversa, associando agli uomini un salario più basso a parità di ore.[3]

- **Discriminazione illegale**, che non è spiegabile o giustificata da attributi non sensibili. Si suddivide a sua volta in:

    - **Discriminazione diretta**: se utilizza direttamente informazioni sensibili per generare risultati sfavorevoli nei confronti di categorie protette

    - **Discriminazione indiretta**: se non utilizza direttamente attributi sensibili ma i risultati sono un loro effetto implicito. Un algoritmo di recruiting potrebbe non aver accesso al sesso di un candidato ma potrebbe inferirlo attraverso altre informazioni disponibili, come la partecipazione ad un programma per sole donne.[4]

[2] Mujtaba e Mahapatra, «Fairness in AI-Driven Recruitment».
[3] Holzinger et al., *Towards Explainability for AI Fairness*.
[4] Mehrabi et al., «A Survey on Bias and Fairness in Machine Learning».

Per evitare queste forme di discriminazione è fondamentale per chi sviluppa questi sistemi considerare la *fairness* a partire dai primissimi stadi di realizzazione, con un approccio multidisciplinare che prenda in considerazione anche aspetti etici, giuridici e sociali.

2.2. Sorgenti Unfairness

I bias umani possono essere trasferiti ai modelli di AI in diverse modalità, tra cui:

- **Dataset Biased**: Se i dataset utilizzati durante l'addestramento, contengono, come spesso accade, *bias*, questi verranno incorporati e propagati dai modelli. Un dataset è considerato *biased* quando presenta un forte sbilanciamento tra le categorie, con gruppi sottorappresentati, oppure quando è stato redatto manualmente e può quindi presentare errori o pregiudizi soggettivi.

- **Definizioni dell'etichetta**: La definizione del target può influenzare il livello di *fairness* dell'algoritmo. Talvolta una etichetta rappresenta un concetto molto più ampio e vago e la sua assegnazione varia a seconda di chi la effettua. L'utilizzo di criteri di selezione che presentano un forte sbilanciamento per categorie protette può falsare le predizioni del modello. Per esempio, un modello di recruiting potrebbe voler identificare "buoni candidati" per una data posizione lavorativa. Tuttavia, esistono diversi fattori, in parte soggettivi, che definiscono un buon candidato. L'utilizzo di determinate caratteristiche, come per esempio la permanenza di un candidato all'interno della azienda, potrebbe penalizzare determinate categorie e perpetuare il problema. L'attributo in esame non riflette necessariamente la bontà di un candidato, in quanto una durata più breve può avere cause diverse. Le donne storicamente tendono ad avere una permanenza più bassa, per motivi che possono essere legati a maternità o discriminazione. Persone con disabilità potrebbero avere permanenze più brevi se in azienda non sono presenti strutture adeguate.

- **Selezione feature**: La selezione delle feature da utilizzare durante l'addestramento può migliorare le performance di un modello, riducendo la complessità computazionale e i tempi di calcolo. Tuttavia, questo può portare a disuguaglianze nel caso in cui gli attributi selezionati non presentino una adeguata rappresentazione per tutte le categorie presenti. Un algoritmo di *recruiting* potrebbe per esempio utilizzare l'Università di provenienza e favorire candidati appartenenti all'università

da cui provengono gli attuali dirigenti. Nel caso l'università in questione non fosse frequentata da candidati appartenenti a categorie protette questo potrebbe influenzare l'algoritmo nel fare correlazioni errate, scartando candidati altrettanto qualificati.

- **Proxy**: Anche se un modello non avesse accesso ad attributi sensibili, potrebbe comunque imparare ad inferirli attraverso altre feature correlate, dette Proxy. Un esempio è dato dall'algoritmo di recruiting sviluppato da *Amazon* che, pur non includendo il genere, aveva imparato a dedurlo attraverso attributi come l'università di provenienza, che poteva talvolta essere riservata ad un solo genere.

- **Masking**: La manipolazione volontaria dei dati per occultare la presenza di *bias*, per esempio attraverso l'aggregazione dei dati in categorie generiche, viene detta *Masking*.[5]

2.3. Bias Detection

Per garantire un trattamento equo nell'ambito dell'assunzione *AI driven* è importante implementare metodi per identificare i *bias* appresi dai modelli, in modo da poterli mitigare.

Un sistema di AI può essere considerato equo se garantisce gli stessi risultati per individui la cui unica differenza è costituita da attributi sensibili.

Negli anni sono state sviluppate diverse definizioni formali per verificare post-processing il livello di equità di un algoritmo. La scelta della metrica più adeguata dipende dal contesto specifico, nel quale falsi positivi e falsi negativi possono avere impatti diversi.

Alcune tra le metriche più diffuse sono:

- **Equalized Odds**: il tasso di veri positivi (TPR) e quello di falsi positivi (FPR) deve essere uguale, tra gruppi diversi identificati da attributi sensibili. Questo significa che il modello deve avere accuratezza uniforme, sia nel caso di classificazioni corrette che errate. Nel contesto di recruiting questo significa che:

[5] Mujtaba e Mahapatra, «Fairness in AI-Driven Recruitment».

- Candidati idonei, appartenenti a categorie diverse, devono avere la stessa probabilità di essere classificati correttamente;
- Candidati non idonei, con competenze insufficienti, devono anch'essi avere la stessa probabilità di essere erroneamente classificati come positivi.

$$P(\hat{Y} = 1|A = 0, Y = y) = P(\hat{Y} = 1|A = 1, Y = y), y \in \{0,1\}$$

(A attributo sensibile, $\hat{Y}$ classe predetta, Y classe reale)

Studi indipendenti hanno dimostrato che il modello COMPAS, utilizzato nel sistema giudiziario degli Stati Uniti, per valutare il rischio di recidiva dei detenuti, non rispetta questa metrica.[6]

- **Equal Opportunity**: rilassamento di Equalized Odds, tiene conto solo del tasso di veri positivi (TPR) e garantisce perciò che campioni di gruppi diversi abbiano uguali probabilità di essere classificati positivamente se appartengono realmente alla classe positiva. Può essere utile quando i falsi positivi sono meno critici dei falsi negativi, come per esempio nell'ambito del recruiting.

$$P(\hat{Y} = 1|A = 0, Y = 1) = P(\hat{Y} = 1|A = 1, Y = 1)$$

- **Demographic Parity**: assicura che il tasso di classificazioni positive (TP + FP) sia sempre lo stesso tra gruppi diversi, senza concentrarsi sulla reale classe di ciascun campione, ignorando quindi l'accuratezza del modello. Questa metrica non tiene conto della reale distribuzione delle classi all'interno dei gruppi diversi, che potrebbe essere biased.

$$P(\hat{Y}\,|A = 0) = P(\hat{Y}\,|A = 1)$$

[6] Patalay, «COMPAS : Unfair Algorithm?»

- **Fairness Through Awareness**: questa metrica richiede che individui simili, definiti attraverso una specifica metrica di similarità, abbiano risultati simili, a prescindere dalla categoria protetta di appartenenza. In questo caso individui con capacità e competenze simili dovrebbero avere la stessa probabilità di *matching* con una posizione lavorativa.

- **Fairness Through Unawareness**: richiede che il modello non utilizzi esplicitamente caratteristiche sensibili nel processo di decision making. Escludendo tali attributi il modello non è in grado di discriminare sulla base di essi. Tuttavia, questa metrica ha dei limiti poiché esistono attributi correlati che possono ricondurre a quelli sensibili e non è pertanto adatta in dataset complessi. Esclude la discriminazione diretta ma non quella indiretta.

- **Treatment Equality**: la proporzione tra falsi negativi e falsi positivi deve essere costante tra gruppi diversi. Questa metrica è particolarmente appropriata in contesti in cui l'entità di una classificazione errata cambia molto a seconda che sia positiva o negativa. Per esempio, nel caso di una banca rifiutare una richiesta di mutuo (FN) o approvarlo (FP) ha un impatto molto diverso e dovrebbe essere bilanciato equamente tra gruppi demografici diversi.

- **Test Fairness**: richiede che il punteggio di probabilità S predetto da un modello (S = S(x)) abbia lo stesso significato predittivo per qualsiasi gruppo. Per ogni valore di S la classe predetta deve essere la stessa.

$$P(Y = 1|S = s, R = b) = P(Y = 1|S = s, R = w)$$

Questo criterio, apparentemente ovvio, è fondamentale in ambiti come la giustizia penale, nei quali è importante che le previsioni abbiano lo stesso significato, indipendentemente da differenze culturali.

- **Counterfactual Fairness**: questa metrica impone che il risultato di un campione rimanga invariato cambiando il valore di un attributo sensibile**.** Un candidato idoneo dovrebbe rimanere idoneo se in un mondo ipotetico avesse un sesso o una etnia diversa. Può essere più complessa da elaborare poiché richiede la rielaborazione da parte del modello di nuovi campioni.

$$P(\hat{Y}_{A\leftarrow a}(U) = y|X = x, A = a) = P(\hat{Y}_{A\leftarrow a'}(U) = y|X = x, A = a)$$

- **Fairness in Relational Domains**: cattura la struttura relazionale del dominio, prendendo in considerazione oltre agli attributi anche le connessioni sociali e organizzative tra gli individui.

- **Conditional Statistical Parity**: estende *Demographic Parity* richiedendo la parità tra gruppi diversi all'interno di specifici sottogruppi, identificati da attributi non sensibili. Garantisce che individui che condividono caratteristiche rilevanti, come per esempio l'area di studio o il livello di esperienza, siano trattati equamente, indipendentemente da attributi sensibili.

Queste metriche forniscono strategie rigorose per ottenere algoritmi più equi ma è importante specificare che solo in rari casi possono essere contemporaneamente soddisfatte. Per questo è indispensabile identificare le metriche più appropriate, sulla base di considerazioni etiche e legali, che siano il più possibile allineate a valori e priorità degli utilizzatori coinvolti.

2.4. Bias Mitigation

La mitigazione delle disparità nei sistemi di AI è una sfida complessa e sfaccettata. Sono stati proposti negli anni vari approcci per affrontare il problema e sono attualmente disponibili diversi toolkit che gli sviluppatori possono utilizzare per esaminare, segnalare e

mitigare la discriminazione nei modelli. Tra questi ultimi i progetti opensource *AIF360*[7] e *Fairlearn*[8].

Questi strumenti accompagnano gli sviluppatori durante tutto il ciclo di vita dei modelli, includendo metriche per dataset e modelli, *explainer* e algoritmi di mitigazione.

Le principali strategie si suddividono in tre categorie, a seconda dello stadio della pipeline a cui vengono applicate:

- **Pre-processing**: se è possibile modificare i dati di training, rimuovendo *bias* o bilanciando classi sottorappresentate.
- **In-processing**: se è possibile modificare la procedura di addestramento.
- **Post-processing**: se l'algoritmo può solo essere trattato black box, senza poter modificare dati o addestramento.[9]

2.4.1. Pre-processing

Un approccio comune è quello di rielaborare i dati di training per garantire che rappresentino l'intera popolazione, inclusi gruppi storicamente emarginati.

Questo può prevedere tecniche di *reweighting*, *oversampling*, *undersampling* e *data augmentation*, attraverso la generazione di dati sintetici. In questa fase possono inoltre essere esclusi attributi non rappresentativi per tutte le categorie presenti nel dataset. Questa strategia implica l'identificazione e la correzione dei bias nei dati prima che il modello venga addestrato su di essi.

Sebbene questa tecnica si sia dimostrata efficiente nel mitigare i bias nei dataset, essa non rimuove i pregiudizi insiti di un modello e potrebbero quindi essere necessari accorgimenti aggiuntivi durante l'addestramento.

[7] Aasheim e Hufthammer, «Bias Mitigation with AIF360: A Comparative Study».
[8] Bird et al., «Fairlearn: A Toolkit for Assessing and Improving Fairness in AI».
[9] Bellamy et al., «AI Fairness 360».

2.4.2. In-processing

A differenza della tecnica precedente i metodi in-processing prevedono la selezione dei modelli più adeguati e l'ottimizzazione di questi ultimi durante la fase di training, imponendo requisiti specifici di *fairness*.

L'ottimizzazione può essere realizzata attraverso l'applicazione di vincoli o con scelte di design specifiche.

Un algoritmo potrebbe per esempio essere addestrato per massimizzare *l'accuracy parity*, garantendo che il tasso di candidati qualificati assunti sia lo stesso tra gruppi demografici diversi.

Un altro approccio, recentemente proposto, è l'*adversarial debiasing*, nel quale la rete principale (*predictor*) massimizza l'*accuracy* delle sue predizioni e contemporaneamente minimizza l'*accuracy* dell'avversario (*adversary*). La rete avversaria ha come obiettivo la predizione degli attributi sensibili sulla base delle predizioni del *main predictor.*

Con queste tecniche il modello è costretto a tenere conto di gruppi sottorappresentati per ottenere performance migliori.

Tuttavia, queste tecniche di *debiasing* sono emerse solo di recente e necessitano di maggiore ricerca per dimostrare la loro fattibilità ed efficienza.

Inoltre, non sempre è possibile alterare la struttura di un modello, che potrebbe in alcuni casi essere un sistema preesistente già addestrato. In questo caso l'unica tecnica applicabile è l'approccio *black box* in *post-processing*.

2.4.3. Post-processing

Le tecniche *post-processing* si focalizzano nella modifica dell'output di modelli già addestrati, per soddisfare specifici requisiti di fairness.

Questo potrebbe per esempio includere la correzione delle soglie di classificazione, talvolta tramite la definizione di soglie separate, sulla base delle metriche precedentemente menzionate.

Il vantaggio di questo approccio è la possibilità di operare *black box*, senza modificare l'architettura sottostante. In aggiunta spesso sono incluse, in questa tecnica, spiegazioni delle predizioni, che evidenziano gli attributi più influenti e favoriscono la trasparenza.

Nell'ambito del recruiting fornire una spiegazione dei risultati a candidati sottoposti ad una selezione tramite AI è cruciale per giustificare risultati sfavorevoli e permettere un miglioramento futuro. È emerso infatti che fornire riscontri chiari può migliorare significativamente la percezione di giustizia da parte dei candidati, durante il processo di *recruiting*.[10]

Tuttavia, anche questo approccio prevede dei limiti, tra cui la riduzione delle performance. Inoltre, da solo, non è in grado di mitigare *bias* derivanti da dati storici o *dataset* sbilanciati.

2.5. Limiti

Nonostante i progressi nelle tecniche di identificazione e mitigazione dei *bias*, il problema dell'*AI fairness* presenta ancora diversi limiti.

In primo luogo, uno dei principali ostacoli è dato dalla scelta di una o più metriche, che può variare significativamente tra applicazioni diverse. La selezione di una metrica rispetto ad un'altra può richiedere un compromesso tra *performance* ed equità. Inoltre, all'aumentare del numero di metriche imposte, il livello di *fairness* potrebbe aumentare ma la complessità computazionale e le prestazioni del modello potrebbero risentirne.

In aggiunta, nell'ambito del *recruiting*, ogni posizione lavorativa è unica, con requisiti specifici, che possono includere attributi sensibili. Un incarico in ambito religioso potrebbe, per esempio, necessitare di dettagli come la religione o il genere.[11] Questo rende difficile

[10] Gilliland, «The Perceived Fairness of Selection Systems».
[11] Mujtaba e Mahapatra, «Fairness in AI-Driven Recruitment».

escludere a priori l'utilizzo degli attributi sensibili per l'addestramento dei modelli di recruiting.

Anche a livello normativo, nonostante l'Unione Europea stia sviluppando normative specifiche con l'*AI Act*, è necessaria una collaborazione tra giurisdizioni internazionali diverse, in quanto i sistemi di AI operano spesso a livello globale. Inoltre, alcune normative già in vigore, come il GDPR in Europa, limitano l'utilizzo dei dati sensibili, essenziali per monitorare il livello di fairness.

Infine, alcune regolamentazioni, come lo stesso *AI Act*[12] e la *AI Bias Audit Law di New York*[13], contengono ambiguità che permettono alle aziende di non effettuare controlli rigorosi se dichiarano le proprie applicazioni come a basso rischio. Queste normative distinguono infatti le applicazioni in applicazioni ad alto rischio, che operano in settori critici ed effettuano decisioni determinanti, e applicazioni a basso rischio, in cui il modello viene utilizzato esclusivamente come supporto decisionale.

Questo solleva preoccupazioni sulla reale efficacia delle normative, in quanto il semplice coinvolgimento umano è sufficiente a bypassare gli *audit* previsti, ma non esclude di per sé la presenza di discriminazione.

[12] «Why the AI Act Fails to Protect Civic Space and the Rule of Law».
[13] «New York Lawmakers Aim to Close Loopholes in NYC's AI Bias Audit Law and Add Teeth to Workplace Protections».

3. Dataset Akkodis

Akkodis è un'azienda globale che si occupa di consulenza e offre servizi di *recruiting* e formazione. Dal 2022 fa parte del gruppo Adecco, leader a livello mondiale nel reclutamento di personale. Questa azienda si distingue nel settore per il suo impegno verso l'inclusività.

"At Akkodis we celebrate diverse backgrounds and empower unique voices. Our inclusive workplace fosters sense of belonging, broad perspectives, and innovation. With our conscious inclusion training and mentorship programs we foster an equitable workforce where everyone can thrive. Join us and make a difference."[14]

Véronique Rodoni

Group SVP HR Akkodis

Il suo approccio inclusivo si riflette anche sui suoi dati, che potrebbero essere meno influenzati da pregiudizi umani, rispetto ai dataset tradizionalmente utilizzati. I risultati ottenuti evidenziano come, utilizzando a monte un dataset più equo, si riduca la necessità di mitigazione durante l'addestramento e in *post-processing.*

3.1. Struttura del Dataset

Il dataset dell'azienda, precedentemente anonimizzato, presenta più righe per ciascun candidato memorizzato.

Per ogni candidato ciascuna riga identifica un diverso step nel processo di recruiting. Le colonne possono essere suddivise in tre macrocategorie: **attributi del candidato**, **attributi del processo di recruiting** e **attributi della posizione** lavorativa, associata al candidato.

[14] «Akkodis.com».

ATTRIBUTI del CANDIDATO:

- **ID**: identificatore univoco
- **Candidate State**: stato del candidato
    - *Imported:* candidati importati da database esterni, come per esempio Alma Laurea. I candidati che mantengono questo stato potrebbero non aver mai risposto ad Akkodis. Alcuni presentano l'evento (*Event_Type__Val*) *CV Request*, che indica che il recruiter non ha ancora ricevuto il curriculum.
    - *First contact:* primi contatti con il candidato, normalmente per telefono. I candidati che mantengono questo stato potrebbero aver interrotto i contatti con Akkodis o potrebbero non avere un curriculum adeguato (*Event_Feedback = Inadequate CV*).
    - *In selection:* candidati in fase di selezione, sottoposti ai primi colloqui, per una posizione lavorativa selezionata tra quelle gestite dalla azienda
    - *QM:* candidati sottoposti a Qualification Meeting
    - *Economic Proposal*: candidati che hanno ricevuto una proposta economica da parte della azienda
    - *Vivier:* candidati le cui competenze non sono risultate allineate con i requisiti della posizione per cui sono stati valutati, ma sono ritenute valide da Akkodis per opportunità future.
    - *Hired* il candidato è stato assunto dalla azienda che ha ingaggiato Akkodis
- **Age Range**: colonna categorica contenente range di età [< 20], [20 – 25], [26 – 30], [31 – 35], [36 – 40], [40 – 45], [> 45]
- **Residence**: attuale residenza del candidato
- **Sex**: sesso del candidato, ammette due valori (*Male* | *Female*) e il valore di default è *Male*.

- **Protected Category**: indica se il candidato appartiene alle categorie protette, specificando l'articolo di riferimento (*articoli 1* e *18*).
- **TAG**: parole chiave utilizzate dal recruiter.
- **Study Area**: area di studio, disciplina accademica del candidato.
- **Study Title**: laurea o titolo accademico conseguito
    - *Middle school diploma*
    - *Professional qualification*
    - *High school graduation*
    - *Three-year degree*
    - *Five-year degree*
    - *Master's degree*
    - *Doctorate*
- **Years Experience**: range di anni di esperienza del candidato [0], [0-1], [1-3], [3-5], [5-7], [7-10], [+10]
- **Sector**: settore nel quale il candidato ha esperienza.
- **Last Role**: ultimo ruolo lavorativo o di studio del candidato.
- **Year of Insertion**: anno di inserimento del candidato nel database.
- **Year of Recruitment**: anno di assunzione del candidato, presente solo se *Candidate State = Hired.*
- **Current Ral**: RAL attuale del candidato.
- **Expected Ral**: aspettativa del candidato sulla RAL futura.

ATTRIBUTI del PROCESSO:

Sono relativi ad uno specifico stadio e cambiano per uno stesso candidato man mano che va avanti nel processo di *recruiting*.

- **Event_Type__Val**: specifica il tipo di evento, lo stadio del processo di reclutamento. Gli eventi presenti nel database possono essere suddivisi in 3 macrocategorie:
    - **Eventi iniziali**: *Commercial note, CV Request, Contact note, Research Association*
    - **Eventi centrali**: *HR interview, BM interview, Technical interview, Qualification Meeting*
    - **Eventi finali**: *Candidate notification, Sending SC to customer, Economic proposal, Notify candidate, Inadequate CV*
- **Event_Feedback**: feedback associato ad uno specifico evento (*Event_Type__Val*), può essere OK o KO, con eventuali commenti specificati tra parentesi. Non tutti i tipi di eventi prevedono un feedback.
- **Overall**: punteggio associato al colloquio, presente solo per righe contenenti *Event_Type__Val* centrali, associati a colloqui.
- **Akkodis headquarters**: sede di Akkodis che gestisce il candidato.

Punteggi assegnati dal recruiter, da 1 a 4, durante il colloquio:

- **Technical Skills**: competenze tecniche.
- **Standing/Position**: posizione all'interno dell'organizzazione.
- **Communication**: competenze comunicative.
- **Dynamism**: livello di dinamicità.
- **Mobility**: mobilità.
- **English**: livello di inglese.

ATTRIBUTI della POSIZIONE LAVORATIVA:

Questi campi sono presenti solo se il candidato è stato assunto dalla azienda in questione.

- **Recruitment Request**: richiesta della azienda per un candidato.
- **Assumption Headquarters**: sede della posizione lavorativa.
- **Job Family Hiring**: categoria della posizione lavorativa.
- **Job Title Hiring**: titolo specifico della posizione.
- **Job Description**: descrizione del ruolo.
- **Candidate Profile**: profilo ideale del candidato, richiesto dalla azienda.
- **Years Experience.1**: anni di esperienza richiesti, espressi in range compatibili con il campo *Years Experience* del candidato.
- **Minimum Ral**: RAL minima prevista.
- **Ral Maximum**: RAL massima prevista.
- **Study Level**: livello di studio richiesto per la posizione lavorativa, i valori sono compatibili con il campo *Study Title.*
- **Study Area.1**: specifica ambito di studio richiesto, contiene compatibili con *Study Area.*
- **Linked_search_key**: campo contenente un codice, non univoco, in formato *RSnn.nnnn*, nel quale il primo numero a due cifre identifica l'anno di inserimento della posizione lavorativa mentre il numero dopo il punto indica il numero di ricerche effettuate per una specifica posizione.

3.2. Preprocessing

3.2.1. Overview

Il dataset presenta circa 12 300 candidati, di cui solo il 4% ha come *Candidate State Hired* e presenta quindi le informazioni riguardanti la posizione lavorativa associata. Dopo una analisi più approfondita è emerso che più del 50% dei candidati è composto da *entry* importate da database esterni, per le quali mancano numerosi campi (*Candidate State = Imported*). La maggior parte di questi candidati, infatti, non ha avuto contatti diretti con Akkodis. Questa distribuzione dei dati ha reso necessaria una fase di preprocessing preliminare per uniformare i dati.

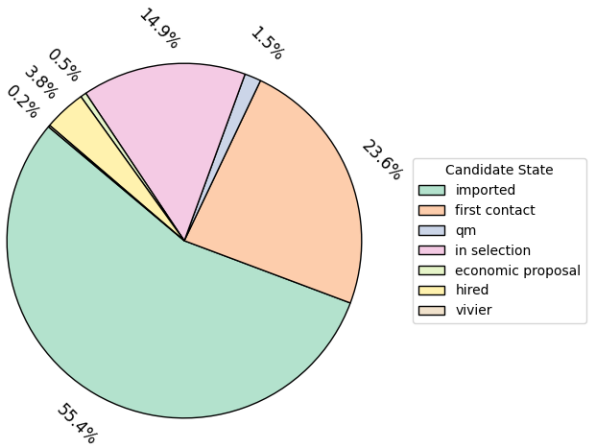

Figura 1 Distribuzione Candidate State nel dataset iniziale

3.2.2. Data Cleaning

Durante questa prima fase è stata analizzata la distribuzione dei valori mancanti, per una comprensione migliore della struttura iniziale del dataset. Questa analisi ha permesso una distinzione chiara tra le tre principali categorie di attributi presenti:

- Attributi relativi al candidato
- Attributi relativi al processo di *recruiting*
- Attributi relativi alla posizione lavorativa

Successivamente tutti i valori mancanti sono stati temporaneamente sostituiti con il valore "*not specified*", per preservare la struttura originale senza introdurre distorsioni. I valori numerici sono stati convertiti nel formato corretto, mentre sono stati trasformati attraverso operazioni statistiche solo nelle fasi successive.

Per garantire una maggiore uniformità dei campi testuali è stato applicato un *mapping* per sinonimi e varianti lessicali, correggendo anche eventuali errori di digitazione.

3.2.3. Feature Mapping

Alcuni campi sono stati rielaborati per estrarre informazioni utili e semplificarne il contenuto.

Protected Category

Il campo *Protected Category* conteneva l'indicazione dell'articolo di riferimento (*articolo 1 | articolo 18)* per i candidati appartenenti a categorie protette. Questo campo è stato rielaborato in una variabile binaria (*yes | no).*

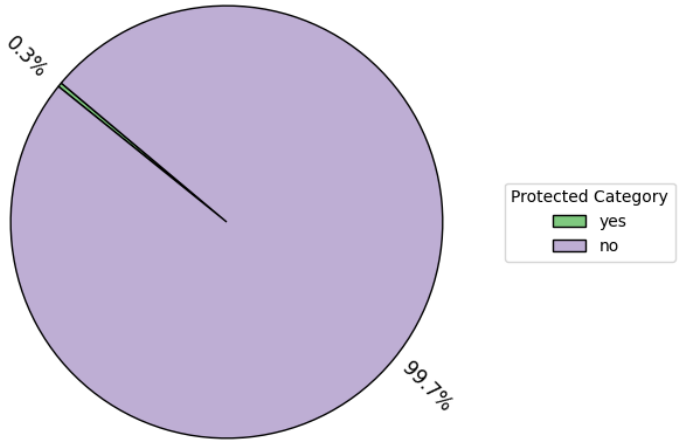

Figura 2 Distribuzione di Protected Category nel dataset iniziale

Residence

Il campo *Residence* conteneva dati non strutturati:

```
['turin » turin ~ piedmont' 'conversano » bari ~ puglia'
'caserta » caserta ~ campania' ...
'san felice a cancello » caserta ~ campania'
'perdifumo » salerno ~ campania'
'palmanova » udine ~ friuli venezia giulia']
```

Figura 3 Contenuto iniziale della colonna Residence

Per rendere l'informazione più organizzata sono state estratte quattro nuove colonne:

Residence Country, Residence Italian Region, Residence Italian Province, Residence Italian City.

Id	Residence	Residence Country	Residence Italian Region	Residence Italian Province	Residence Italian City
15	squinzano » lecce ~ puglia	italy	puglia	lecce	squinzano
36	alessandria » alessandria ~ piedmont	italy	piedmont	alessandria	alessandria
39	bari » bari ~ puglia	italy	puglia	bari	bari
41	perch dosimo » cremona ~ lombardy	italy	lombardy	cremona	perch dosimo
47	germany » (state) ~ (overseas)	germany	not in italy	not in italy	not in italy
...	...	...	...	...	...

Figura 4 Dettaglio colonne Residence, Residence Country, Residence Italian Region, Residence Italian Province, Residence Italian City

È probabile che il campo *Residence* sia stato sottoposto a traduzione automatica, per questo alcune città estratte risultano inesistenti. Questa anomalia viene trattata nelle fasi successive. Sono inoltre state introdotte due colonne binarie aggiuntive: *Italian Residence* ed *European Residence.*

Last Role

La colonna Last Role conteneva numerose varianti terminologiche per posizioni simili. Per ridurre la frammentazione e garantire consistenza è stato applicato un mapping per uniformarla.

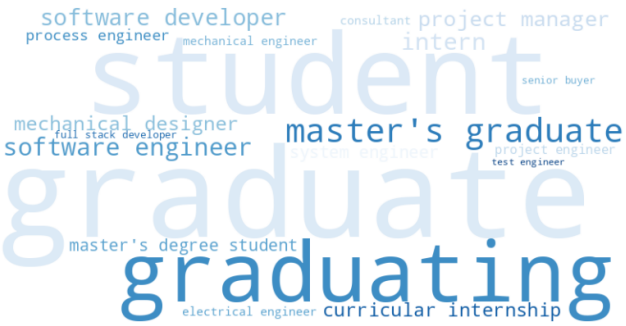

Figura 5 Cloud Word dei top 20 Last Role nel dataset iniziale

3.2.4. ID duplicati

Il database fornito da **Akkodis** è frutto della fusione di due database diversi, appartenenti alle due aziende AKKA e Modis, delle quali è stata effettuata una fusione nel 2022 da parte di Adecco. [15]

Per questo motivo alcuni ID risultano duplicati e appartengono a candidati diversi all'interno del dataset. Un campo utile per l'identificazione di ID duplicati è dato da *Year of Insertion*, che rappresenta l'anno di inserimento del candidato all'interno del database e dovrebbe quindi essere costante.

Il processo di assegnazione di un nuovo ID è stato realizzato sulla base delle informazioni fornite da Akkodis, confrontando i valori di campi invarianti, come *Year of Insertion, Sex, Age Range etc.*

[15] Adecco Group, «The Adecco Group completa l'acquisizione della quota di maggioranza di AKKA Technologies».

L'azienda proprietaria ha affermato che nessuno dei suoi dipendenti ha mai richiesto un cambio di sesso all'interno del database. Tuttavia, risultano più di 300 ID per i quali il campo *Sex* assume più di un valore. Questa incongruenza può avere due possibili motivazioni:

- Durante la fusione candidati diversi con sesso diverso hanno mantenuto lo stesso ID;
- Non è possibile lasciare il campo *Sex* non specificato qualora sconosciuto dal recruiter e il valore di default previsto è *male.*

Durante questa fase sono stati generati più di 1.000 nuovi ID, portando il numero totale di candidati distinti da 12.300 a 13.350.

3.2.5. Dataset Sorting

Per ogni variabile categorica è stato definito un ordine, sia per motivi di visualizzazione che per riordinare il dataset. In particolare:

- **Candidate State**: l'ordine assegnato coincide con quello fornito da Akkodis, secondo l'ordinamento interno di recruiting

  '*imported' > 'first contact' > 'in selection' > 'qm' > 'vivier' > 'economic proposal' > 'hired'*

- **Age Range**: *'< 20 years' > '20 - 25 years' > '26 - 30 years' > '31 - 35 years' > '36 - 40 years' > '40 - 45 years' > '45 years'*

- **Years Experience**: *'not specified' > '[0]' > '[0-1]' > '[1-3]' > '[3-5]' > '[5-7]' > '[7-10]' > '[+10]'*

- **RAL**: *'not specified' > '-20k' > '20-22k' > '22-24k' > '24-26k' > '26-28k' > '28-30k' > '30-32k' > '32-34k' > '34-36k' > '36-38k' > '38-40k' > '40-42k' > '42-44k' > '44-46k' > '46-48k' > '48-50k' > '+50k'*

3.3. Scelta del Target

L'obiettivo del progetto è lo sviluppo di modelli di AI, da impiegare nelle fasi preliminari del processo di *recruiting*, per analizzare i bias che emergono. La scelta del target è stata influenzata dalla struttura del dataset, che contiene i candidati e le posizioni gestiti da Akkodis.
In base ai dati disponibili sono stati identificati due possibili obiettivi per la predizione automatica:

- **RAL**: una nuova colonna per la predizione della RAL più adeguata al profilo del candidato.

- **Hired**: una nuova colonna che etichetta le coppie candidato-posizione come positive o negative, definendo se il profilo del candidato sia adeguato alle richieste della azienda.

La prima ipotesi è stata scartata poiché più del 90% dei candidati non presenta alcun valore per nessuno dei campi associati alla RAL.

D'altra parte, per poter distinguere tra candidati idonei e non idonei, è necessario analizzare le tre principali tipologie di candidati presenti:

- **Candidati nei primi stadi di recruiting**: candidati dei quali non si hanno sufficienti informazioni, poiché ancora nelle fasi preliminari.

- **Candidati valutati per una posizione lavorativa ma non assunti**: in questo caso la posizione lavorativa per la quale sono stati valutati non è presente nel dataset. Questa tipologia si suddivide a sua volta in:

    - **Candidati adeguati** alla posizione lavorativa, che non sono stati assunti per motivazioni indipendenti dalle competenze.

    - **Candidati inadeguati** per la posizione in questione. Questa tipologia può assumere un nuovo valore *Vivier* per *Candidate State,* se il candidato viene considerato potenzialmente valido per offerte future e viene quindi mantenuto all'interno del dataset.

- **Candidati assunti**: candidati in stato *Hired*, possiedono tutte le informazioni relative alla posizione lavorativa.

Per garantire consistenza è necessario rimuovere i candidati che sarebbero etichettati positivamente ma che non presentano la posizione lavorativa associata. Tra questi vi sono i candidati nelle primissime fasi di recruiting, non ancora valutati per una posizione, e i candidati che per motivazioni non correlate non sono andati avanti nel processo di assunzione.

3.3.1. Rimozione Candidati nelle prime fasi di recruiting

Attraverso una analisi dei valori di *Event_Type__Val* e *Event_Feedback* per ogni *Candidate State* è possibile isolare i candidati nelle prime fasi del processo di reclutamento:

- ***Imported***: questo stato rappresenta candidati importati da altri database. Il tipo di evento associato (*Event_Type__Val*) è normalmente *CV Request* o *Contact Note.* Questa tipologia è stata interamente scartata ai fini del progetto.

- ***First Contact***: questo stato rappresenta i candidati contattati da Akkodis, che non hanno ancora effettuato colloqui. Per questo stato sono stati filtrati tutti i candidati con un solo evento iniziale e che non presentavano il campo *Sector*, che è stato scelto come requisito minimo necessario per poter valutare il profilo di un candidato.

- ***In Selection***: questo stato rappresenta candidati in fase di valutazione. Gli eventi più frequenti sono *BM interview* ed *HR interview*. Anche in questo caso vengono filtrati i candidati che presentano solo eventi iniziali.

- ***QM***: questo stato rappresenta candidati che sono stati sottoposti a qualification meeting.

- ***Vivier***: questo stato rappresenta candidati che non sono stati assunti ma che sono stati ritenuti validi da Akkodis per opportunità future.

- ***Economic Proposal***: questo stato rappresenta candidati che hanno ricevuto una offerta economica da parte della azienda. In questo caso nel campo *Event_Feedback*

è presente la ragione per cui il candidato si trovi ancora in questo stato e non in stato *Hired.* Alcuni valori presenti sono ad esempio *'OK (other candidate)', 'KO (proposed renunciation)', 'KO (ral)' etc.*

- ***Hired***: questo stato rappresenta candidati che hanno concluso positivamente il processo di recruiting e sono stati assunti per la posizione lavorativa specificata nei campi appositi.

Durante questa fase sono stati rimossi circa 9.000 candidati.

3.3.2. Rimozione Candidati validi

Per garantire consistenza dei dati è stato necessario rimuovere tutti i campioni che sarebbero stati idonei ma che, per motivazioni indipendenti, non sono stati assunti. Il criterio di selezione è stato l'ultimo feedback ricevuto dal candidato e l'ultimo evento associato.

A questo scopo sono stati identificati tutti gli *Event_Feedback* che evidenziano una valutazione positiva del profilo o un eventuale esito negativo indipendente dalle competenze:

- ***OK (other candidate)***: il candidato non è stato assunto nonostante il riscontro positivo (*OK*), perché qualcun altro è stato selezionato. Non è corretto etichettare negativamente profili che hanno ricevuto questo feedback.

- ***KO (lost availability)****:* il candidato non è più disponibile, non è coerente assegnare una valutazione negativa del suo profilo.

- ***OK (hired)***: il candidato è stato assunto, il database potrebbe non aver aggiornato il suo *Candidate State* qualora questo sia diverso da *Hired.*

- ***OK (waiting for departure)***: il candidato ha ricevuto un esito positivo dalla azienda, l'incarico sta per iniziare.

- ***KO (opportunity closed)***: il candidato non è stato assunto perché l'opportunità lavorativa per la quale era stato contattato non è più disponibile. Le sue competenze non sono la causa del suo *Candidate State.*

- ***KO (retired)***: il candidato si è ritirato.
- ***KO (ral)***: il candidato ha rinunciato alla opportunità per motivi economici.
- ***KO (proposed renunciation)****:* il candidato ha rinunciato alla opportunità.

Sono stati rimossi tutti i candidati che avevano ricevuto come ultimo feedback, nell'ultima riga associata, uno dei precedenti.

Sono stati inoltre filtrati dai restanti candidati tutti coloro che presentavano un evento finale tra i seguenti:

- ***Economic Proposal***: il candidato ha ricevuto una proposta economica dalla azienda; quindi, le sue competenze rispettano i requisiti della posizione.
- ***Candidate Notification***: questo evento rappresenta l'ultima comunicazione con il candidato dopo che l'azienda ha fatto la proposta economica.

Durante questa fase sono stati filtrati circa 500 candidati.

3.4. Creazione dataset finale

Il dataset iniziale fornito da Akkodis contiene solo le posizioni lavorative dei candidati assunti, mentre non vi è alcun dato specificato per i candidati con *Candidate State* diverso da *Hired*. Ai fini dell'addestramento è stato generato un nuovo dataset per ospitare coppie (*candidato, posizione*), etichettate attraverso una nuova colonna binaria *Hired*.

A questo scopo tutte le posizioni lavorative sono state raccolte in un *dataset* ausiliario, che è stato a sua volta sottoposto ad analisi e *pre-processing* per escludere le posizioni duplicate. Ad ogni posizione lavorativa è stato assegnato un ID univoco per semplificare le fasi successive.

Per ogni candidato è stata mantenuta una singola riga rimuovendo le colonne *Event_Type__Val* ed *Event_Feedback* ed effettuando una media numerica per le colonne dei punteggi ottenuti durante colloqui diversi.

Nel dataset finale sono stati aggiunti tutti i candidati assunti, con le relative posizioni lavorative associate, etichettati positivamente (*Hired=1*).

Le coppie negative sono state generate artificialmente attraverso l'utilizzo di una funzione di similarità basata sulla *cosine similarity*, applicata ai vettori generati con *TF-IDF (Term Frequency-Inverse Document Frequency) Vectorizer.* I vettori utilizzati sono stati ottenuti attraverso la congiunzione delle colonne testuali:

- *Tag, Study Area, Sector, Last Role* per il candidato.
- *Recruitment Request, Job Family Hiring, Job Title Hiring, Job Description, Candidate Profile, Study Area.1* per la posizione lavorativa.

Questo approccio ha permesso di selezionare in modo sistematico posizioni più o meno affini al profilo del campione, simulando il reale processo di *recruiting*.

Per ogni candidato assunto è stata quindi introdotta una riga aggiuntiva, etichettata negativamente, con una posizione lavorativa scelta casualmente tra le posizioni meno simili al candidato in questione.

Per ogni candidato non assunto sono state invece aggiunte due righe negative con:

- Una posizione lavorativa selezionata tra le più simili al profilo del candidato, per simulare una reale candidatura.
- Una posizione lavorativa tra le meno compatibili, per introdurre un confronto più netto.

L'utilizzo della funzione di similarità per la selezione delle posizioni simula il reale processo di *recruiting*, nel quale i candidati vengono valutati per posizioni compatibili con il loro profilo. L'utilizzo di un numero ridotto di righe negative per ciascun candidato limita lo sbilanciamento della colonna target e riflette la natura del *dataset*, nel quale la maggior parte dei candidati è stata valutata per una singola posizione tra le disponibili.

Il dataset finale così generato contiene circa 8.400 righe.

3.4.1. Colonne Accessorie

Al dataset ottenuto sono state aggiunte delle colonne ausiliarie, per rappresentare numericamente le relazioni tra gli attributi del candidato e della posizione lavorativa.

Colonne di similarità, basate sulla cosine similarity:

- *Similarity_Score:* Misura la similarità tra il profilo del candidato e la posizione lavorativa, sulla base delle colonne descrittive. Questo valore è stato utilizzato per selezionare le posizioni nella fase di generazione delle coppie negative.

- *Similarity_Score_Last_Role:* Misura la similarità tra l'ultimo ruolo ricoperto dal candidato (*Last Role)* e il titolo della posizione lavorativa (*Job Title Hiring)*, ipotizzando che candidati con una esperienza recente simile siano avvantaggiati.

- *Similarity_Score_Study_Area*: Misura la similarità tra il settore di studio del candidato (*Study Area)* e quello richiesto (*Study Area.1)*.

Colonne di confronto per attributi ordinali:

Per ogni coppia di campi corrispondenti, come per esempio *Study Title* e *Study Level,* è stata definita una scala di livelli ordinati e il punteggio è stato calcolato secondo la formula:

$$score = \frac{CandidateLevel - PositionLevel}{levelMax - levelMin}$$

dove $levelMax$ e $levelMin$ rappresentano rispettivamente i valori massimo e minimo nella scala dei livelli associata agli attributi.

I punteggi così ottenuti sono normalizzati tra -1 e +1:

- Un **valore nullo** indica una corrispondenza perfetta, il candidato possiede i requisiti richiesti.

- Un **valore positivo** indica che il candidato possiede requisiti superiori a quelli richiesti.

- Un **valore negativo** indica che il candidato possiede requisiti insufficienti rispetto a quelli richiesti.

Le colonne derivate includono:

- *Study_Title_Score*: Misura la corrispondenza tra il titolo di studio del candidato (*Study Title*) e il livello richiesto dalla posizione (*Study Level*).

- *Years_Exp_Score*: Misura la corrispondenza tra gli anni di esperienza del candidato (*Years Experience*) e quelli richiesti (*Years Experience.1*).

- *Ral_Score*: Confronta le aspettative salariali del candidato con la RAL minima offerta dalla posizione lavorativa. In questo caso la formula prevede la differenza tra il livello della RAL offerta dalla azienda e quello del candidato. Un valore negativo indica che la RAL richiesta dal candidato è superiore alla minima offerta dalla posizione, che è quindi insufficiente rispetto alle aspettative.

Colonna per la distanza geografica:

- *Distance (km)*: Rappresenta la distanza tra la residenza del campione e la sede della azienda (*Assumption Headquarters).* È stata ottenuta utilizzando due dataset ausiliari, *italy_geo.xlsx* [16]e *countries.csv* [17],contenenti le coordinate di latitudine e longitudine rispettivamente per le città italiane e per i paesi esteri. In questo contesto tutte le città italiane sono state tradotte attraverso un mapping *ad hoc*. Dopo aver calcolato le coordinate corrispondenti la distanza è stata calcolata con la funzione *geodesic* della libreria *Geopy*.

[16] HenryChinaski, «Comuni-Italiani-2018-Sql-Json-excel».
[17] «Countries.csv».

4. Addestramento

4.1. Preprocessing

Il dataset generato nelle fasi precedenti è stato sottoposto ad ulteriore preprocessing, per garantire dati corretti e adatti all'addestramento dei modelli nelle fasi successive.

Riduzione delle colonne

Sono state rimosse colonne ridondanti, non strutturate o non disponibili nelle prime fasi del processo di selezione, per evitare *data leakage* e ridurre la dimensionalità del dataset.

Alcuni esempi:

- *Candidate State*: Stato finale del candidato, non disponibile prima del processo di selezione, al momento dell'utilizzo di un modello di predizione.
- *Job Description*: Campo troppo esteso e strutturato, utilizzato durante l'elaborazione della colonna *Similarity_Score*.
- *Years Experience.1*: Campo rappresentato attraverso la colonna *Years_Exp_Score*, che lo mette in relazione con *Years Experience* del candidato.

```
columns_to_drop = ['Candidate State', 'Tag' , 'Recruitment Request', 'Last Role',
                   'Job Description', 'Candidate Profile', 'Year Of Insertion', 'Year Of Recruitment',
                   'Job Title Hiring', 'Years Experience.1', 'Study Level', 'Study Area.1',
                   'Job ID', 'Residence_coord', 'Assumption_coord', 'Assumption Headquarters', 'Residence Italian City',
                   'Candidate_Info', 'Position_Info', 'Akkodis Headquarters', 'Job Family Hiring', 'Linked_Search__Key']
```

Figura 6 Colonne rimosse in fase di pre-processing

Encoding degli Attributi Categorici

Le colonne categoriche sono state convertite in formato numerico, attraverso *encoding*, per permettere il loro utilizzo da parte dei modelli di machine learning.

Le variabili ordinali, come *Age Range*, sono state mappate attraverso una scala numerica coerente con il loro significato.

Le variabili categoriche nominali sono invece state convertite utilizzando il `LabelEncoder` di *Scikit-Learn*, che assegna un numero intero univoco a ciascuna categoria.

Valori mancanti

I valori numerici mancanti sono stati sostituiti con la media della rispettiva colonna. I valori categorici hanno invece mantenuto *'not specified'* come categoria a parte, codificata anch'essa nella fase precedente.

Standardizzazione delle feature numeriche

Le seguenti colonne numeriche sono state standardizzate, utilizzando lo `StandardScaler` di *Scikit-Learn*, che utilizza la media e la deviazione standard di ciascuna colonna per trasformare i suoi valori.

```
columns_to_standardize = ['Years Experience', 'Minimum Ral', 'Ral Maximum', 'Current Ral',
                          'Expected Ral', 'Distance (km)']

scaler = StandardScaler()
```

Figura 7 Colonne standardizzate in fase di pre-processing

Questo garantisce che tutti i campi numerici siano alla stessa scala e abbiano quindi lo stesso impatto sul modello.

4.2. Analisi del dataset

Prima dell'addestramento è stata effettuata una analisi grafica delle principali feature, con enfasi sugli attributi sensibili e sulla loro relazione con altri attributi rilevanti.

4.2.1. Matrice di correlazione

Per evidenziare le relazioni tra le colonne del dataset e ottenere una rappresentazione grafica riassuntiva è stata generata una matrice di correlazione, utilizzando la funzione `heatmap` di *Seaborn.*

L'immagine, in Figura 8, evidenzia alcune relazioni, tra cui:

- La colonna **target** *Hired* mostra una correlazione con *Similarity_Score;*

- *Years Experience* evidenzia correlazioni attese con *Current Ral, Expected Ral* e *Technical Skills*, *Years_Exp_Score* e *Age Range_encoded;*

- I punteggi ottenuti durante i colloqui sono correlati tra loro: *Overall, Technical Skills, Standing/Position, Communication, Maturity, Dynamism;*

- La RAL minima prevista dalla azienda *Minimum Ral* mostra una correlazione con *Years_Exp_Score*, evidenziando come il suo valore sia correlato agli anni di esperienza richiesti dalla azienda;

- I campi RAL del candidato *Current Ral* ed *Expected Ral* sono anch'essi correlati con *Ral Score*, che rappresenta la relazione tra le aspettative del candidato e la proposta economica della azienda;

- Il campo *Distance (km)* mostra correlazioni attese con i campi relativi a residenza: *European Residence_encoded*, *Italian Residence_encoded.*

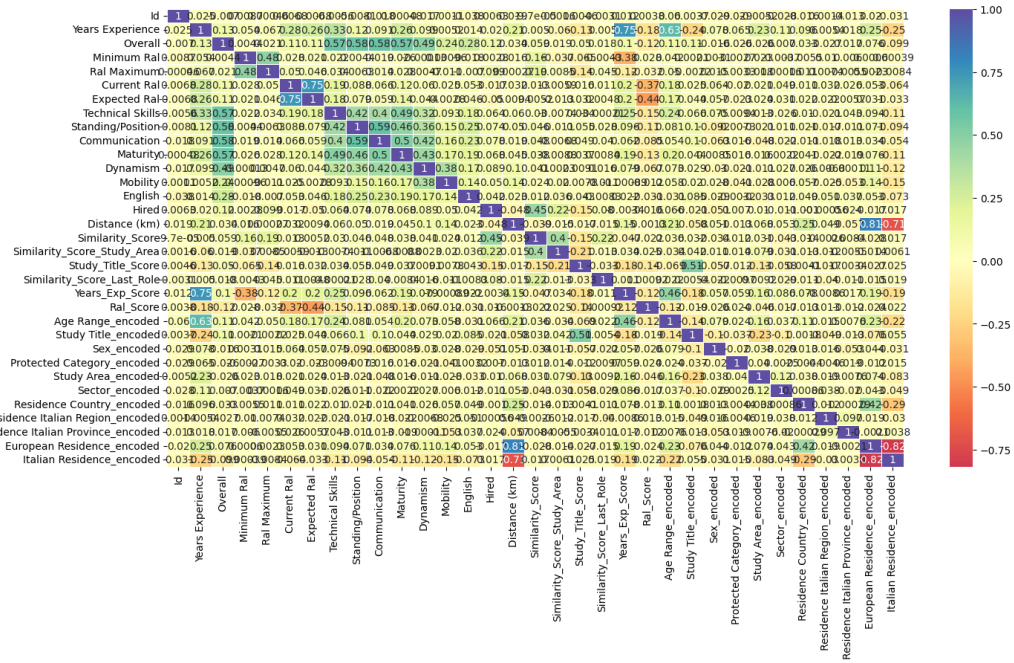

Figura 8 Correlation Matrix

4.2.2. Colonna Target

La distribuzione finale della colonna target, rappresentata dal grafico a torta in Figura 9, risulta molto sbilanciata, evidenziando come solo il 6% delle *entry* sia etichettata positivamente, attraverso il campo *Hired*.

Sarà necessario, in fase di addestramento, bilanciare il dataset attraverso campioni sintetici, per garantire che i modelli non predicano solo classi negative.

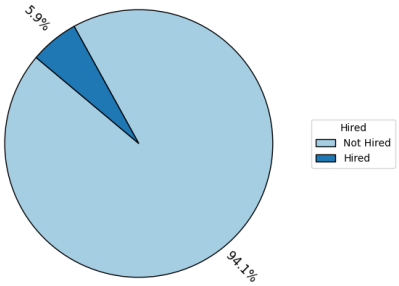

Figura 9 Distribuzione Hired

4.2.3. Analisi Attributi Sensibili

È stata analizzata la distribuzione di attributi sensibili e la loro relazione con campi determinanti come la colonna **target** *Hired*, i titoli di studio *Study Title_encoded*, gli anni di esperienza *Years Experience,* la RAL massima prevista *Ral Maximum* etc.

Sex

Il dataset risulta molto sbilanciato, con solo il 20% dei candidati appartenenti alla categoria *female*, evidenziando una sottorappresentazione.

Tuttavia, analizzando la distribuzione dell'etichetta *Hired* all'interno di ciascun sottogruppo di *Sex* emerge che, in percentuale, le donne sono state assunte maggiormente. In particolare, risulta che l'8,3% delle donne è stato assunto contro il 5,3% per gli uomini.

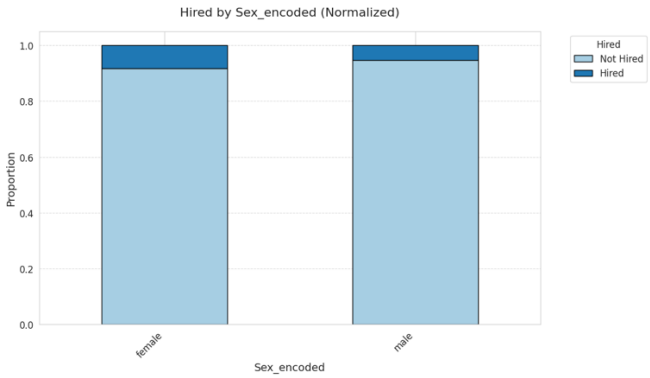

Figura 10 Distribuzione Hired normalizzata per ogni categoria (female|male)

Questa leggera differenza potrebbe però trovare giustificazione nelle competenze dei candidati, come è possibile notare dalla distribuzione dei titoli di studio delle due categorie, in Figura 11. Le donne del dataset tendono ad avere infatti titoli di studio più importanti, come *five-year degree, master's degree* e *doctorate.*

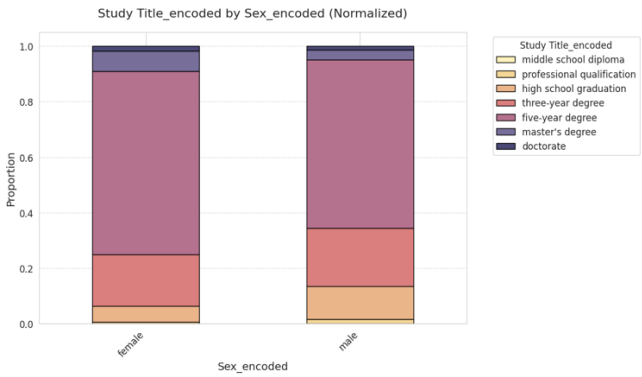

Figura 11 Distribuzione Study Title_encoded *normalizzata per ogni categoria (female|male)*

Tuttavia, analizzando i valori associati alla colonna *Ral Maximum,* per i candidati assunti, emerge una differenza in contrasto: a parità di titolo di studio (*Study* Title) e anni di esperienza (*Years* Experience) le donne tendono ad avere retribuzioni (*Ral Maximum*) più basse (Figura 12, Figura 13).

D'altra parte, non avendo informazioni sufficienti sul numero di ore o sul tipo di contratto, non è possibile giustificare né condannare queste differenze.

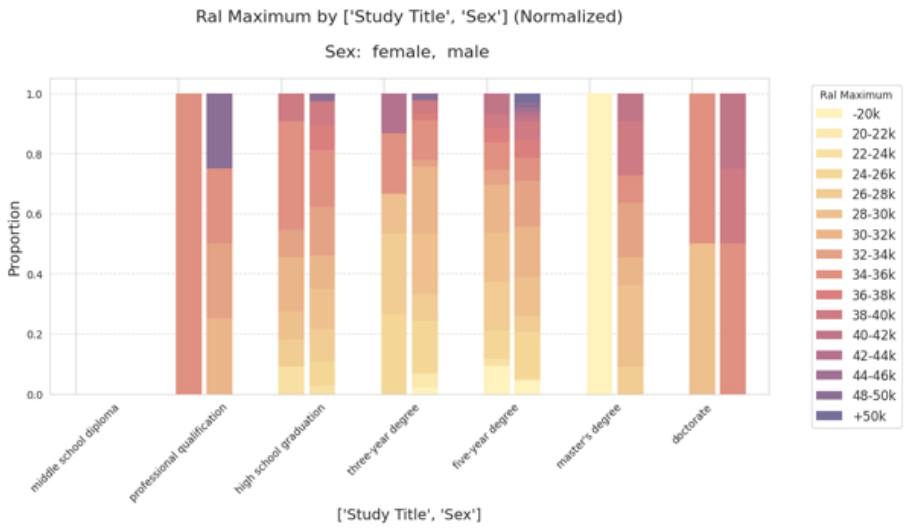

Figura 12 Confronto Ral tra sottogruppi di Sex
a parità di Study Title

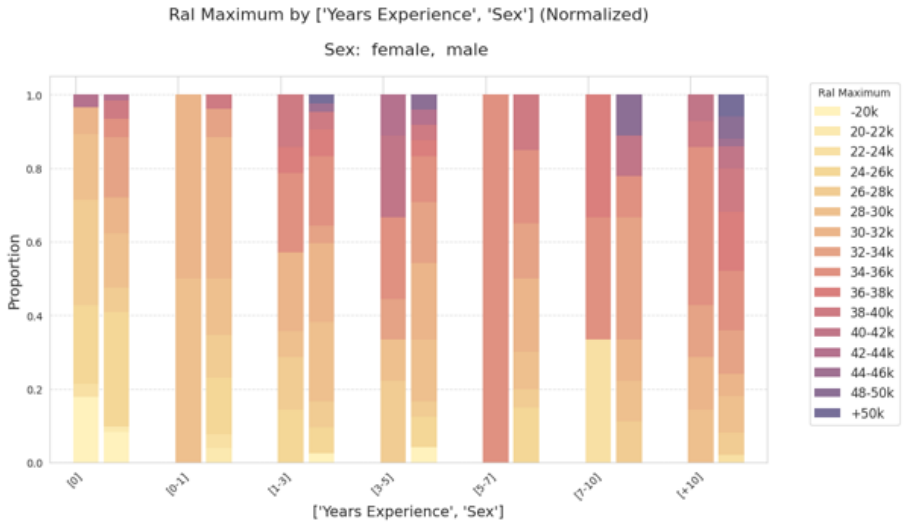

Figura 13 Confronto Ral tra sottogruppi di Sex
a parità di Years Experience

Protected Category

Il dataset è fortemente sbilanciato nei confronti del campo *Protected Category*, con solo lo 0.6% composto da candidati appartenenti a categorie protette.

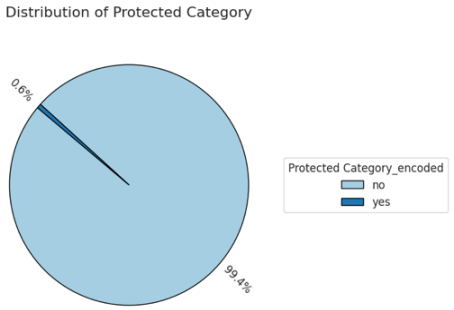

Figura 14 Distribuzione Protected Category

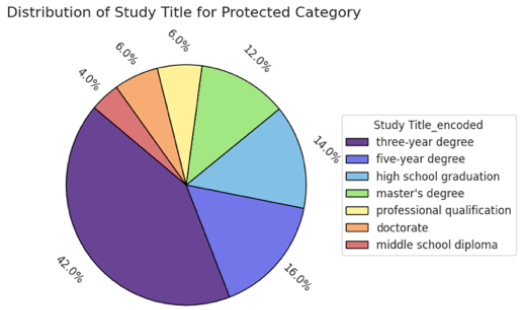

Figura 15 Distribuzione titoli di studio per candidati appartenenti a categoria protetta

Il numero ridotto di candidati appartenenti a categoria protetta, pari a diciotto, non da' la possibilità di effettuare un confronto accurato a parità di competenze.

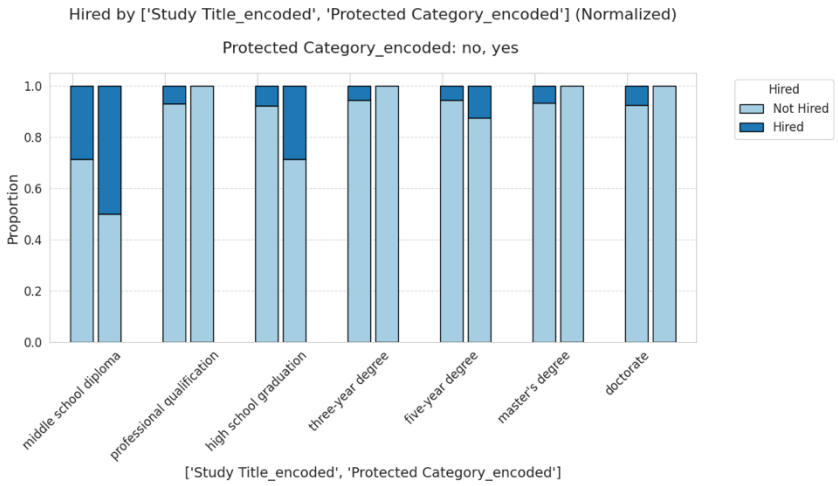

Figura 16 Confronto distribuzione Hired per sottogruppi di Protected Category, a parità di Study Title

Age Range

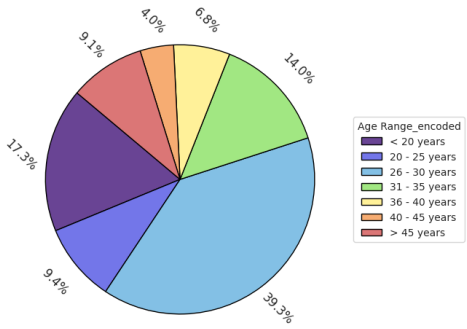

Figura 17 Distribuzione Age Range

Più del 65% del dataset è composto da candidati di età inferiore a 30 anni, con il 17% di età inferiore ai 20. Tuttavia, anche se in numero minore, le restanti categorie hanno un tasso di assunzione più alto.

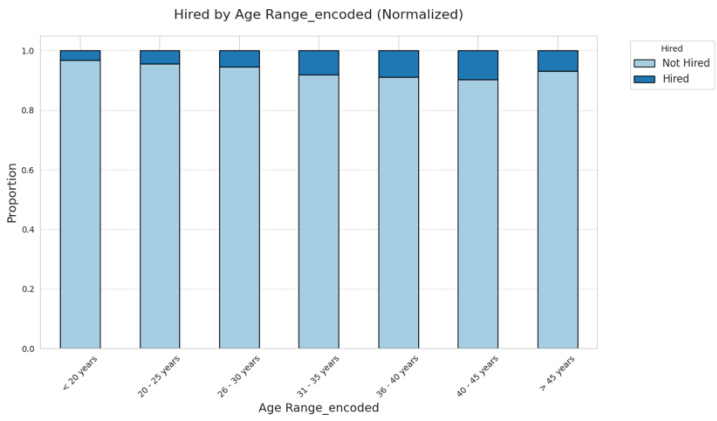

Figura 18 Distribuzione Hired per ogni sottogruppo di Age Range

Residenza

La maggior parte dei candidati risiede in Italia. Tra i candidati con residenza non italiana più del 25% risiede in Oman.

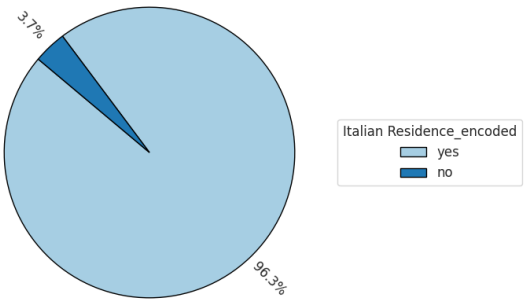

Figura 19 Distribuzione Italian Residence

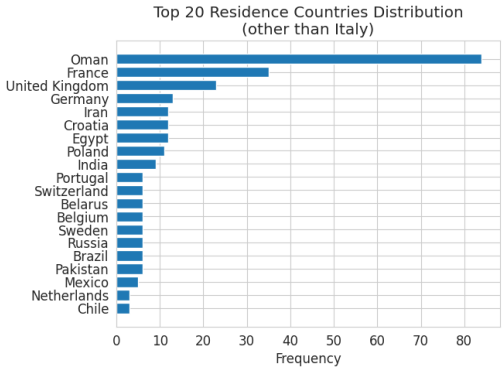

Figura 20 Top 20 dei Paesi di residenza (Italia esclusa)

All'interno del dataset il 6% dei campioni residenti in Italia risulta assunto contro il 4% per i candidati non residenti.

Analizzando però gli altri attributi emerge che i candidati non residenti in Italia hanno in percentuale titoli di studio superiori (*Study Title*) e più anni di esperienza (*Years Experience*). Tuttavia, a parità di titolo di studio, la distribuzione di *Ral Maximum* varia significativamente tra candidati residenti e non residenti.

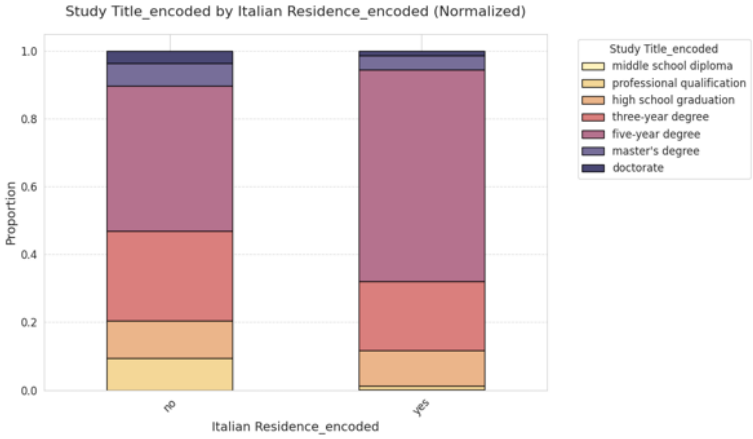

Figura 21 Distribuzione Study Title per candidati non residenti e residenti in Italia

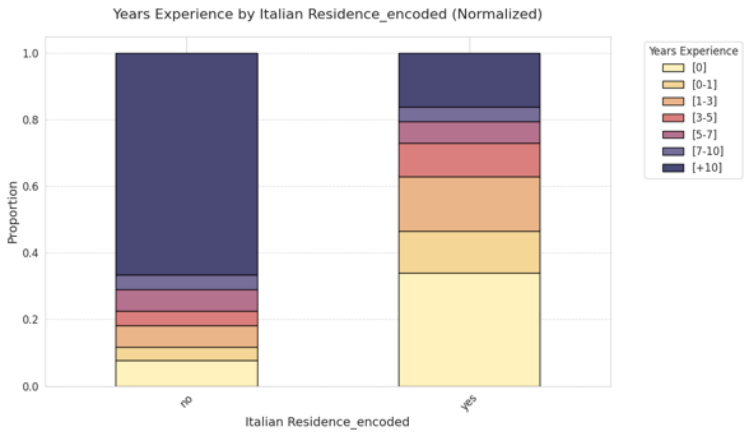

Figura 22 Distribuzione Years Experience per candidati non residenti e residenti in Italia

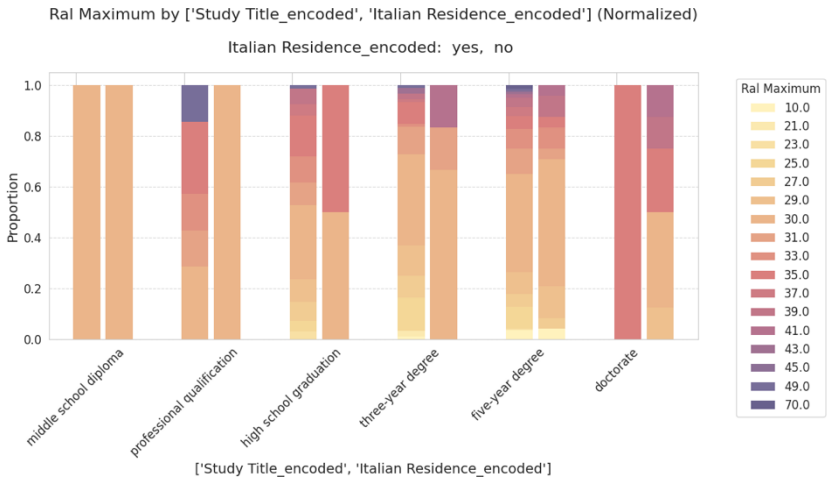

Figura 23 Distribuzione Ral Maximum per candidati residenti e non residenti in Italia, a parità di titolo di studio

4.3. Scelta dei modelli

Per garantire una panoramica complessiva sono stati selezionati modelli appartenenti a famiglie algoritmiche diverse, includendo modelli lineari, probabilistici, ad albero e reti neurali. L'obiettivo è confrontare i risultati e mettere in relazione *performance* e *fairness.*

I modelli selezionati includono:

- *Modelli Lineari*
  - `Linear Regression`
  - `Logistic Regression`
  - `Linear Perceptron`
- *Modelli Probabilistici*
  - `Gaussian Naïve Bayes`
- *Modelli Tree-based*
  - `Decision Tree`
  - `Random Forest`
  - `AdaBoost`
- *Modelli Distance-based*
  - `K-Nearest Neighbors`
- *Reti Neurali*

4.3.1. Rete Neurale

Per poter utilizzare strumenti standard nelle fasi successive, come `GridSearchCV`, la rete neurale è stata realizzata attraverso una classe ad hoc, `CustomKerasClassifier`, che estende `BaseEstimator` e `ClassifierMixin` di *Scikit-Learn*.

Il modello adottato per le reti neurali è una rete *fully connected*, realizzata utilizzando *Tensorflow* e *Keras*.

```
class CustomKerasClassifier(BaseEstimator, ClassifierMixin):
    def __init__(self, input_shape, neurons=10, activation='relu', optimizer='adam',
                 batch_size=32, epochs=20, learning_rate=0.001):
        ...

    def build_tf_model(self):
        model = Sequential([
            Input(shape=self.input_shape),
            Dense(self.neurons, activation=self.activation),
            BatchNormalization(), Dropout(0.4),
            Dense(self.neurons, activation=self.activation),
            BatchNormalization(), Dropout(0.4),
            Dense(self.neurons // 2, activation=self.activation),
            BatchNormalization(),
            Dense(1)
        ])

        optimizer_fn = {'adam': Adam, 'rmsprop': RMSprop, 'sgd': SGD}.get(self.optimizer)
        if optimizer_fn is None:
            raise ValueError("Unknown optimizer")

        model.compile(optimizer=optimizer_fn(learning_rate=self.learning_rate),
                      loss=BinaryCrossentropy(from_logits=True), metrics=['accuracy'])
        return model

    def fit(self, X, y, validation_data=None, early_stopping=None):
        ...
```

Figura 24 Pseudo codice classe CustomKerasClassifier

L'architettura, come mostrato in Figura 24, è composta da:

- Un *input layer* la cui dimensione coincide con il numero di feature del dataset.
- Tre *Dense hidden layer*, in cui la funzione di attivazione e il numero di neuroni sono parametri da ottimizzare durante la *Gridsearch*. Il terzo strato utilizza la metà dei neuroni (*neurons // 2).*
- Ciascun hidden layer è seguito da *Batch Normalization,* per aumentare la stabilità, e *Droupout* al 40% per ridurre l'overfitting.
- L'*output layer* è composto da un singolo neurone, senza funzione di attivazione Sigmoidea. La funzione di perdita utilizzata è la `BinaryCrossentropy(from_logits=True)`, che incorpora la sigmoide in modo più stabile.

L'*optimizer* e il *learning rate* sono anch'essi parametri selezionati nelle fasi successive attraverso la *Gridsearch*. La metrica utilizzata è l'*accuracy.*

Il modello utilizza inoltre il parametro `class_weight` nel metodo `fit()`, per bilanciare i pesi tra le due classi. I pesi vengono calcolati attraverso la funzione `compute_class_weight` con `class_weight = 'balanced'` come parametro.

4.4. Addestramento

Prima dell'addestramento è stata effettuata una ricerca degli iperparametri per ottimizzare le performance dei modelli selezionati, rispettivamente attraverso *Grid Search* e *Randomized Search*, a seconda del numero di parametri coinvolti.

4.4.1. Modelli Tradizionali

Per i modelli tradizionali il dataset è stato suddiviso inizialmente in *training/validation set* e *test set*, utilizzando la funzione `train_test_split` di *Scikit-Learn*. È stato specificato uno split stratificato attraverso il parametro `stratify` per garantire che tutti i set contenessero la stessa distribuzione di campioni per la classe positiva (*Hired = 1*) e la categoria protetta (*Protected Category_encoded*). Per alcuni modelli selezionati è stato inoltre adottato `SelectKBest`, per identificare le 15 feature più rappresentative nel dataset.

Sono state utilizzate `GridSearchCV` e `RandomizedSearchCV`, a seconda del numero di combinazioni di parametri per ciascun modello.

Per realizzare la *cross validation* è stato impiegato `StratifiedKFold` sul *training/validation set.*

I parametri sottoposti a *Gridsearch* per ciascun modello tradizionale, memorizzati in un dizionario, sono i seguenti:

- **Linear Regression**: `fit_intercept`, `positive`
- **Logistic Regression**: `C`, `solver`, `class_weight`, `max_iter`

- **Linear Perceptron**: `early_stopping`, `class_weight`
- **Gaussian Naïve Bayes**: `var_smoothing`
- **Decision Tree**: `max_depth`, `class_weight`
- **Random Forest**: `max_depth`, `n_estimators`, `class_weight`
- **AdaBoost**: `n_estimators`, `learning_rate`
- **K-Nearest Neighbors**: `n_neighbors`, `weights, metric`

Le metriche utilizzate per la ricerca sono *f1_macro, recall_macro, precision_macro, roc_auc,* per garantire la corretta predizione di entrambe le classi. L'*accuracy* non è stata utilizzata poiché non è affidabile in dataset molto sbilanciati, in cui basta predire sempre la classe maggioritaria per ottenere un punteggio alto.

Per ogni modello è stato scelto, tra i quattro selezionati dalle metriche, quello che ne massimizzava la media.

I risultati ottenuti dopo l'addestramento di ciascun modello sono rappresentati dalla *heatmap* in Figura 25.

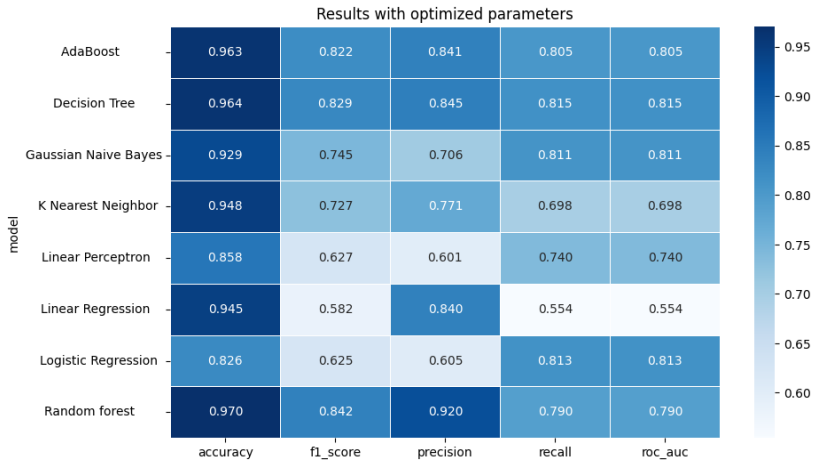

Figura 25 Heatmap metriche dopo Grid / Randomized Search

Come previsto, trattandosi di un dataset molto sbilanciato, la maggior parte dei modelli tende a predire più frequentemente la classe negativa, e per questo presenta *accuracy* elevata, ma *recall* relativamente bassa rispetto a *precision*.

I parametri ottenuti per ciascun modello sono mostrati in Figura 26.

```
DecisionTreeClassifier(max_depth=14, random_state=42)

LinearRegression()

LogisticRegression(C=100, class_weight='balanced', max_iter=500,
                   random_state=42)

GaussianNB(var_smoothing=1e-05)

RandomForestClassifier(max_depth=20, min_samples_leaf=2, n_estimators=200, random_state=42)

Perceptron(early_stopping=True, random_state=42)

KNeighborsClassifier(metric='manhattan', n_neighbors=2, weights='distance')

AdaBoostClassifier(learning_rate=1.5, n_estimators=40, random_state=42)
```

Figura 26 Parametri ottenuti per i modelli tradizionali

4.4.2. Rete Neurale

L'addestramento della rete neurale è stato effettuato su un *dataset* bilanciato rispetto alla colonna target *Hired* attraverso `CTGAN`, per ottimizzare le prestazioni.

Per mantenere un *validation set* con soli candidati reali, non coinvolti nell'addestramento di `CTGAN`, il *train/validation set* è stato suddiviso esplicitamente in *training set* e *validation set*. Questa scelta permette di utilizzare il set di validazione nella *cross validation,* senza rischio di *data leakage* e il set di addestramento per generare campioni sintetici.

I campioni sintetici ottenuti sono stati aggiunti al *training set* originale, per ottenere un dataset bilanciato `balanced_train_set` nei confronti del target.

Successivamente, il `balanced_train_set` è stato unito al validation set per eseguire la *cross-validation*.

Per la *cross-validation*, è stato utilizzato `StratifiedKFold`, per definire *fold* stratificati del test di validazione, composti solo da candidati reali. I *training fold*, sono stati poi integrati con i campioni del `balanced_train_set`, aumentando così il numero di campioni disponibili per l'addestramento.

```
n_splits = 5
strat_kfold = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=random_state)

train_splits = []
val_splits = []

for train_index, val_index in strat_kfold.split(val_set, val_set['Hired']):
    train_index = train_index.tolist() + balanced_train_set.index.tolist()

    val_splits.append(val_index.tolist())

    train_splits.append(train_index)

cv_splits = [(train_splits[i], val_splits[i]) for i in range(n_splits)]
```

Figura 27 Utilizzo di StratifiedKFold per la generazione di fold ad hoc

Ricerca dei parametri

La selezione dei parametri è stata effettuata attraverso `RandomizedSearchCV`, per ridurre il costo computazionale e valutare un maggior numero di configurazioni.

I parametri esplorati sono stati:

- Numero di neuroni **neurons** per strato `(64, 128, 256, 512)`
- **Funzione di attivazione** `(relu, elu, tanh)`
- **Ottimizzatore** `(adam, rmsprop, sgd)`
- **Dimensione del batch** `(16, 32, 64, 128)`
- **Numero di epoche** `(15, 30, 50, 100)`
- **Learning rate** `(0.001, 0.01, 0.1)`

È stata selezionata *f1_weighted* come metrica per la *Randomized Search*, per bilanciare *precision* e *recall*.

I parametri ottenuti al termine, con un valore di *f1_score* superiore a 0.98, sono rappresentati in Figura 28.

```
best_params_nn = {'optimizer': 'rmsprop',
                  'neurons': 256,
                  'learning_rate': 0.001,
                  'epochs': 50,
                  'batch_size': 64,
                  'activation': 'elu'}
```

Figura 28 Parametri selezionati dalla Randomized Search per la rete neurale

Valutazione modelli

Il miglior modello ottenuto con la *Randomized Search*, `grid_result.best_estimator_`, è stato utilizzato per effettuare predizioni sul *test set*.

Oltre al modello selezionato sono state addestrate tre reti neurali aggiuntive, utilizzando i parametri ottimali precedentemente ottenuti, ma con *seed* diversi.

Il confronto tra *training loss* e *validation loss* evidenzia un'elevata variabilità nel *validation set*, segnalando una possibile instabilità nei risultati.

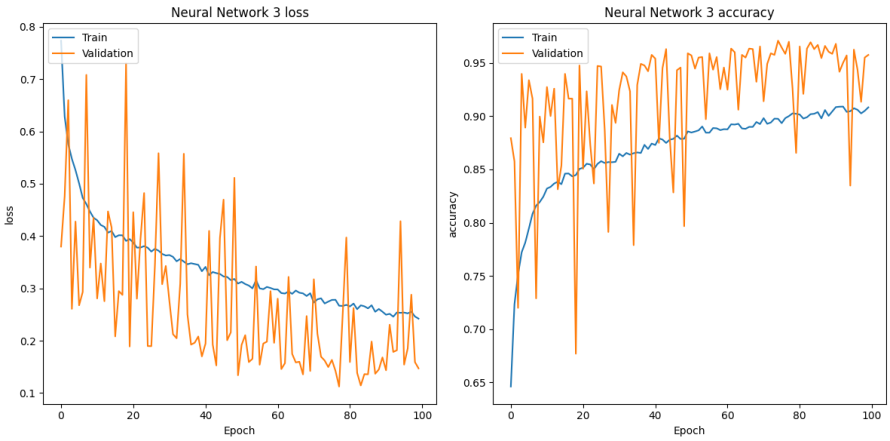

Figura 29 Andamenti Train/Validation Loss/Accuracy per la terza rete neurale

Tuttavia, i modelli addestrati evidenziano prestazioni competitive, come mostrato nella *heatmap* seguente (Figura 30):

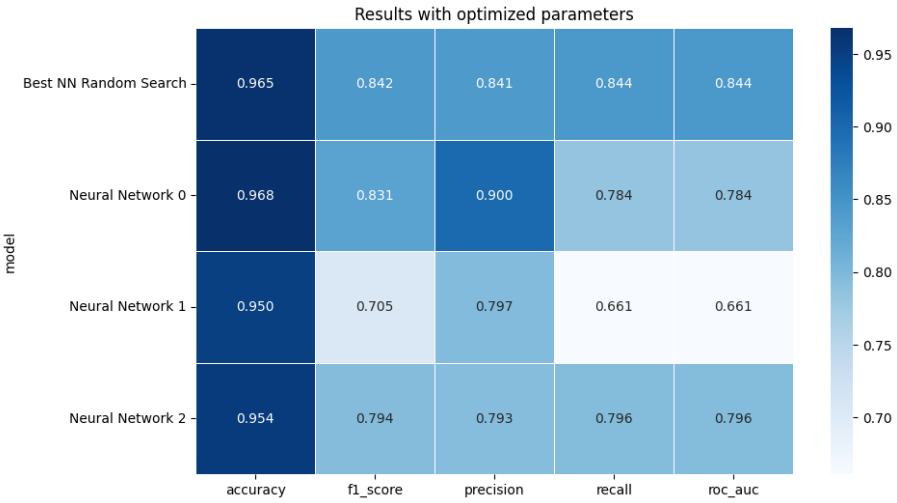

Figura 30 Performance delle reti neurali addestrate

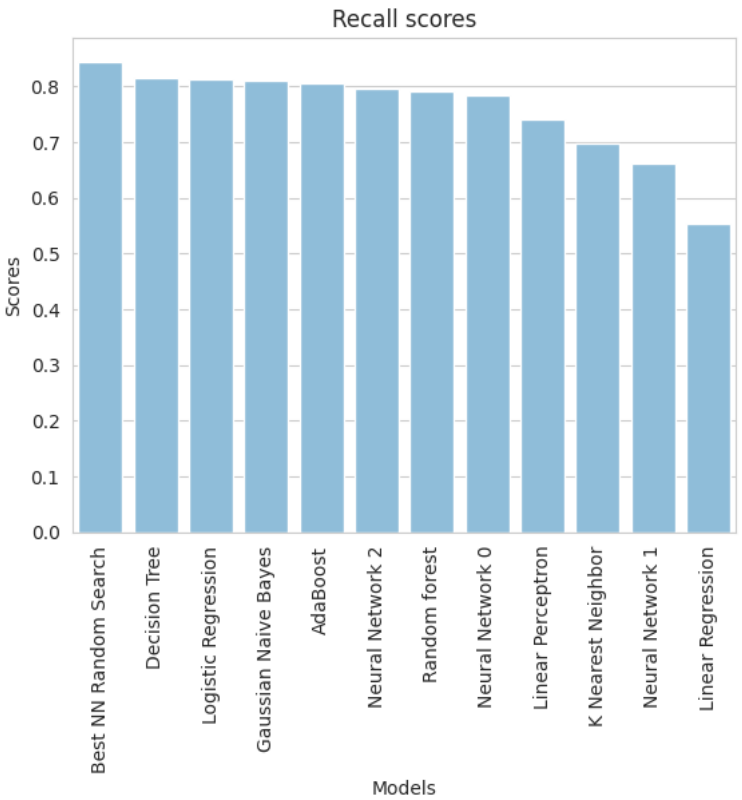

Figura 31 Istogramma della Recall per ciascun modello

5. Analisi dei Bias

5.1. Metriche scelte

Per valutare il livello di *Fairness* dei modelli addestrati sono state scelte tre metriche:

- **Demographic Parity**: Questa metrica richiede che ciascun gruppo abbia le stesse opportunità di essere assegnato alla classe positiva (*Hired=1*), a prescindere che si tratti di un vero o falso positivo. Potrebbe considerare come *unfair* un modello accurato, nel caso di sottogruppi sbilanciati nei confronti della colonna target *Hired* nel *test set*. Per esempio, nel dataset di riferimento le donne hanno un tasso di assunzione più elevato rispetto agli uomini. Ipotizzando che il test set abbia la stessa distribuzione, un modello accurato potrebbe essere considerato *unfair* secondo questa metrica. Tuttavia, in caso in cui il test set presenti sbilanciamenti per l'etichetta *Hired* causati da bias, questi vengono identificati se perpetuati dal modello.

- **Equalized Odds**: Questa metrica garantisce che i tassi di *True Positive* e *False Positive* siano costanti tra gruppi diversi. Questo significa, per esempio, che il modello dovrebbe erroneamente classificare come positivi candidati non idonei con uguale probabilità per individui appartenenti a categorie diverse, senza favorire nessun sottoinsieme.

- **Counterfactual Fairness**: Questa metrica richiede che il modello rieffettui le predizioni su un *test set* alternativo, in cui è stato cambiato solo il valore di un attributo sensibile. Se per esempio cambiando il sesso di tutti i candidati da *female* a *male* il numero di predizioni positive aumenta, questo significa che a parità di profili il modello favorisce campioni con *Sex=male.*

5.2. Implementazione Metriche

Le tre metriche selezionate sono state implementate attraverso tre funzioni ad hoc. È stata inoltre implementata una funzione accessoria per gestire l'*output* di queste ultime e restituire

un *dataframe Pandas*, in cui ciascuna colonna rappresenta un attributo sensibile e ciascuna riga rappresenta uno dei modelli implementati.

5.2.1. Demographic Parity

La funzione `calculate_demographic_parity` implementa la metrica omonima, verificando per ciascun sottogruppo dell'attributo sensibile specificato, il tasso di previsioni positive.
Per attributi sensibili binari, come *Sex*, verifica che la differenza tra i due gruppi sia sotto la soglia di tolleranza specificata.

Nel caso invece di attributi sensibili multi-classe, come *Age Range*, utilizza un test del chi-quadrato per verificare la dipendenza tra predizioni e attributo sensibile. In questo caso verifica che il *p-value* sia superiore alla soglia di *significance level*.

Se il modello risulta *unfair* la funzione restituisce il sottogruppo che risulta discriminato, che in questo caso corrisponde al valore con una minor percentuale di predizioni positive; in alternativa restituisce *True*.

Come previsto la funzione, inFigura 32, non prende in considerazione le etichette reali del test set `y_test`.

```
def calculate_demographic_parity(predictions, X_test, sensitive_feature, tolerance):

    df = pd.DataFrame({'predictions': predictions, 'sensitive_attribute': X_test[sensitive_feature]})
    positive_proportions = df.groupby(sensitive_feature)['predictions'].mean()
    percentage_difference = positive_proportions.max() - positive_proportions.min()

    if percentage_difference > tolerance:
        return positive_proportions.idxmin()
    return True
```

Figura 32 Pseudo codice calculate_demographic_parity, caso binario

5.2.2. Equalized Odds

La funzione `calculate_equalized_odds` calcola *True Positive Rate* (TPR) e *False Positive Rate* (FPR) per ogni sottogruppo identificato dall'attributo sensibile ricevuto tra gli

argomenti. Per entrambi i valori verifica che la differenza tra il valore massimo e il minimo sia sotto la soglia di tolleranza. Se la differenza supera `tolerance` la funzione restituisce il sottogruppo discriminato insieme a TPR e FPR associati, altrimenti *True.*

In questo caso i sottogruppi discriminati sono quelli per cui il modello ha predetto un minor rate di veri positivi o di falsi positivi.

```
def calculate_equalized_odds(predictions, true_labels, sensitive_feature, tolerance):

    metrics = {group: compute_TPR_FPR(predictions[sensitive_feature == group],
                                      true_labels[sensitive_feature == group])
               for group in unique(sensitive_feature)}

    max_tpr_diff = max(TPR_values(metrics)) - min(TPR_values(metrics))
    max_fpr_diff = max(FPR_values(metrics)) - min(FPR_values(metrics))

    if max_tpr_diff <= tolerance and max_fpr_diff <= tolerance
        return True
    return discriminated_groups
```

Figura 33 Pseudo codice calculate_equalized_odds

5.2.3. Counterfactual Fairness

La funzione `calculate_counterfactual_fairness` implementa la metrica omonima. In particolare, per ogni sottogruppo identificato dall'attributo sensibile specificato, crea un *test set* fittizio `X_counterfactual` nel quale sostituisce tutti i valori dell'attributo sensibile con il valore del sottogruppo in esame. Il nuovo set viene utilizzato per effettuare nuove predizioni, attraverso `model.predict`, confrontate con le predizioni precedenti. Per ogni sottogruppo viene calcolata la consistenza e la differenza in percentuale di predizioni positive `delta`.

Se la differenza tra `delta_max` e `delta_min` tra le classi dell'attributo sensibile è sotto la soglia specificata `tolerance` e la consistenza rilevata è maggiore di `(1-tolerance)` la funzione restituisce *True.*

Viceversa, nel caso in cui il modello si riveli *unfair,* vengono restituiti i gruppi discriminati, insieme al *delta* associato.

```
def calculate_counterfactual_fairness(predictions, model, sensitive_feature, X_test, tolerance):
    consistency = {}
    impact_delta = {}

    for group in unique(sensitive_feature):
        X_counterfactual = X_test.copy()
        X_counterfactual[sensitive_feature] = group

        new_preds = model.predict(X_counterfactual)
        new_preds = binarize(new_preds)

        consistency[group] = 1 - mean(abs(predictions - new_preds))
        impact_delta[group] = (sum(new_preds) - sum(predictions)) / len(predictions)

    if check_fairness(consistency, impact_delta, tolerance):
      return True
    return discriminated_group
```

Figura 34 Pseudo codice calculate_counterfactual_fairness

5.3. Bias Detection

Demographic Parity

Tutti i modelli addestrati non rispettano *Demographic Parity* (Figura 36) per *Age Range*. Questo risultato potrebbe essere dovuto allo sbilanciamento della classe *Hired* tra i sottogruppi di *Age Range*.

Il sottogruppo discriminato è identificato da *Age Range = '<20 years'*, che nel dataset fornito presenta il minor tasso di assunzioni, come visto in Figura 18.

Analizzando la distribuzione tra le sottocategorie di *Age Range* nel **test set** emerge una differenza ancora più marcata, come mostrato in Figura 35, con solo l'1,4% dei candidati sotto i 20 anni, assunti.

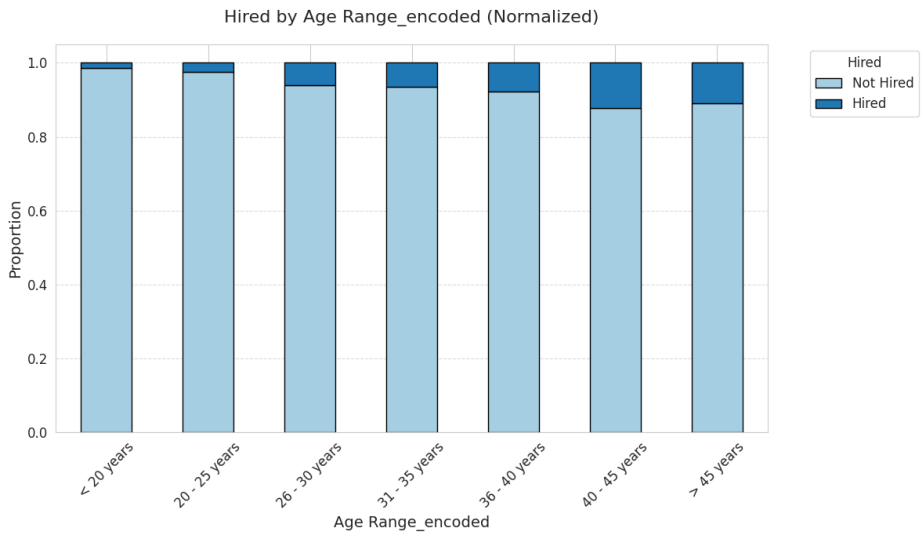

Figura 35 Distribuzione Hired nel test set, per sottogruppi di Age Range

Questi risultati possono quindi essere giustificati dall'accuratezza del modello, che predice correttamente le disparità già presenti nel test set, che potrebbero essere causate da pregiudizi o da anni di esperienza insufficienti.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		< 20 years			
Linear Regression		< 20 years			
Logistic Regression		< 20 years		non-european	
Gaussian Naive Bayes		< 20 years			
Random forest		< 20 years			
Linear Perceptron		< 20 years	no	non-european	
K Nearest Neighbor		< 20 years			
AdaBoost		20 - 25 years			
Neural Network 0		< 20 years			
Neural Network 1		> 45 years			
Neural Network 2		< 20 years			
Best NN Random Search		20 - 25 years			

Figura 36 Demographic Parity

Dall'analisi condotta emerge perciò come questa metrica "costringa" i modelli a bilanciare le categorie protette, anche in caso di test set *biased.* Questo comporta però performance più basse, necessarie per non riflettere le discriminazioni già presenti.

5.3.1. Equalized Odds

Come detto questa metrica si occupa di verificare che la differenza tra i *true* e i *false positive rate* di sottogruppi diversi non superi una soglia fissata `tolerance`.

Come possiamo notare dai risultati ottenuti, in Figura 37, la maggior parte dei modelli tende ad avere tasso di veri positivi molto sbilanciato. I sottogruppi che presentano il TPR minimo sono quelli per cui il modello è meno accurato nella predizione della classe positiva.

Tuttavia, investigando la distribuzione dei sottogruppi nel *test set* emerge che sia per *Protected Category = yes* che per *European Residence = no,* esiste un solo campione assunto (*Hired = 1).* Questo significa che TPR per questi gruppi può assumere solo 0 o 1. Per questo motivo qualsiasi altro sottogruppo avrà in genere un TPR maggiore di 0 o minore di 1.

Il caso di *Italian Residence* è simile, con due soli candidati non italiani assunti.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		< 20 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	no [TPR=0.643]
Linear Regression		< 20 years [TPR=0.0]			
Logistic Regression		26 - 30 years [TPR=0.75]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Gaussian Naive Bayes		26 - 30 years [TPR=0.625]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Random forest		40 - 45 years [TPR=0.375]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Linear Perceptron		< 20 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
K Nearest Neighbor		40 - 45 years [TPR=0.125]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
AdaBoost		40 - 45 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Neural Network 0		26 - 30 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Neural Network 1		> 45 years [TPR=0.176]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Neural Network 2		20 - 25 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	no [TPR=0.612]
Best NN Random Search		20 - 25 years [TPR=0.5]	no [TPR=0.0]	non-european [TPR=0.0]	no [TPR=0.653]

Figura 37 Equalized Odds

5.3.2. Counterfactual Fairness

I risultati di questa metrica, in Figura 38, evidenziano per ogni sottogruppo la differenza percentuale del numero di predizioni positive.

Per esempio, la `Neural Network 2`, per un test set fittizio in cui tutti i candidati hanno come *Sex male,* predice circa lo 0.2% in meno di etichette positive. La stessa rete sembra invece essere più favorevole nei confronti delle donne, con 2.5% in più di predizioni positive, evidenziando un possibile bias all'interno della categoria *Sex*.

Alcuni modelli sembrano discriminare i candidati italiani, principalmente perché favoriscono i candidati non italiani. Questi risultati saranno analizzati nella fase successiva attraverso *explainers* ad hoc.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree					
Linear Regression					
Logistic Regression		< 20 years [-4.93%], > 45 years [8.78%]			
Gaussian Naive Bayes					
Random forest					
Linear Perceptron					
K Nearest Neighbor					
AdaBoost					
Neural Network 0			yes [-0.06%], no [6.05%]		
Neural Network 1			yes [-0.18%], no [19.53%]		
Neural Network 2	male [-0.24%], female [2.55%]	< 20 years [-1.42%], > 45 years [2.67%]	yes [-0.06%], no [15.25%]	european [0.24%], non-european [0.71%]	yes [-1.66%], no [0.0%]
Best NN Random Search			yes [0.18%], no [13.83%]	european [0.06%], non-european [13.59%]	

Figura 38 Counterfactual Fairness

5.4. Explainability

L'*AI explainability* è indispensabile per comprendere meglio il processo decisionale dei modelli, migliorando la trasparenza e la fiducia degli utenti finali.

In questo contesto sono stati utilizzati due *explainers*, LIME *(Local Interpretable Model-agnostic Explanations)* e SHAP *(SHapley Additive exPlanations)* per analizzare gli attributi più influenti per ciascun modello. Attraverso questi strumenti è possibile verificare l'influenza degli attributi sensibili sui modelli e l'impatto di particolari valori sui risultati.

Inoltre, analizzando gli attributi più importanti è possibile identificare discriminazione indiretta nei modelli.

5.4.1. LIME

LIME genera spiegazioni locali per modelli *black-box*, attraverso la generazione di input perturbato per analizzare il comportamento del modello. In questo modo è in grado di

distinguere quali attributi hanno influenzato maggiormente la predizione per uno specifico campione e in che direzione.[18]

Sono state indagate le 20 feature più influenti per ogni cluster del test set identificato da uno specifico attributo sensibile.

Sex

Come anticipato dalle metriche utilizzate nessun modello presenta l'attributo sensibile *Sex* tra gli attributi più influenti. Questo risultato è dovuto al dataset utilizzato, che non presenta sbilanciamenti particolari per la colonna target *Hired* tra sessi diversi.

Tuttavia, tutti i modelli analizzati risultano fortemente influenzati dai campi relativi alla RAL: *Ral Maximum*, *Expected Ral, Minimum Ral*.

Come visto nel paragrafo 4.2.3, queste colonne sono molto probabilmente soggette a bias e la loro influenza potrebbe quindi introdurre **discriminazione indiretta**.

Protected Category

Per i candidati appartenenti a categorie protette è emerso che nessun modello utilizza, tra le 20 feature più influenti, l'attributo sensibile *Protected Category_encoded.*

Questo permette di escludere una possibile discriminazione diretta nei confronti di questo attributo. Tuttavia, anche in questo caso, i risultati mostrano una forte influenza da parte dei campi relativi alla RAL.

[18] Alikhademi et al., «Can Explainable AI Explain Unfairness?»

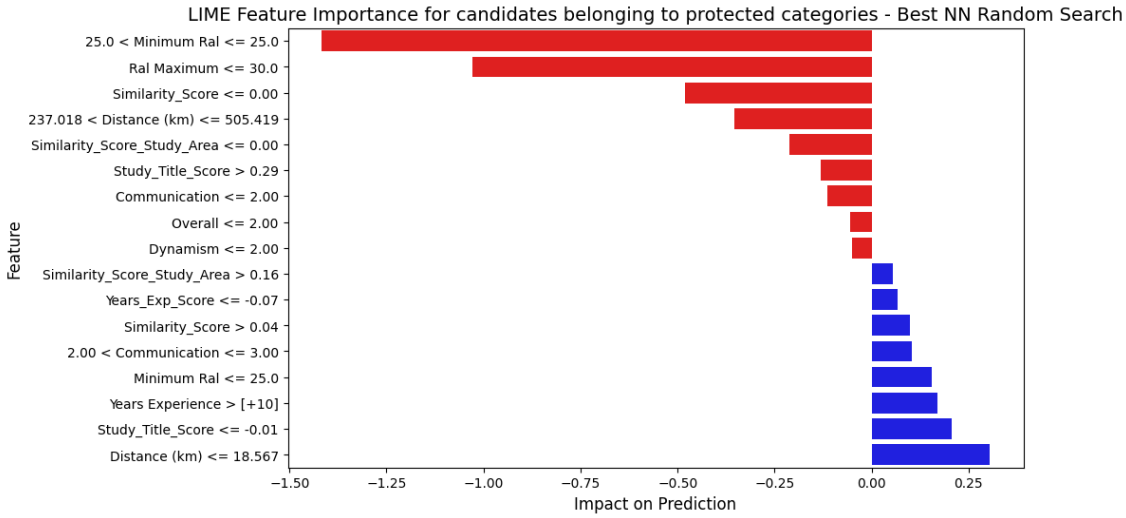

Figura 39 Distribuzione influenza delle 20 feature più importanti per il modello di rete neurale, per i candidati appartenenti a categorie protette

Age Range

Analizzando il comportamento dei modelli per candidati di età inferiore ai venti anni emerge che la maggior parte dei modelli è influenzato dall'attributo sensibile *Age Range*, evidenziando una possibile **discriminazione diretta** nei confronti di candidati molto giovani. Questo risultato è stato evidenziato anche attraverso le metriche di fairness utilizzate nella sezione precedente.

Anche in questo caso le colonne relative a RAL risultano molto influenti.

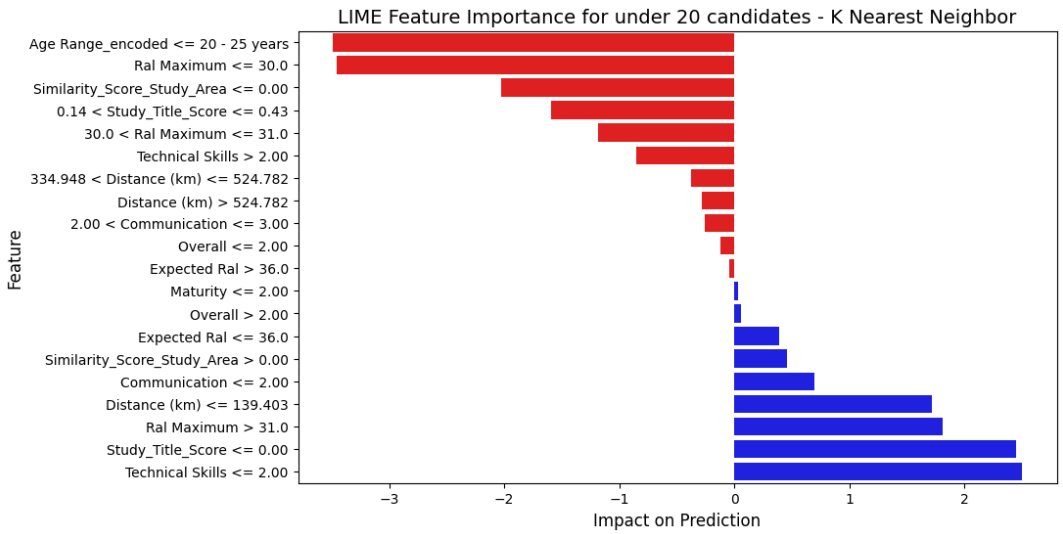

Figura 40 Distribuzione influenza delle 20 feature più importanti per il modello K Nearest Neighbor, per candidati con Age Range = '< 20 years'

Italian Residence

Per i candidati non residenti in Italia nessun modello mostra una particolare influenza da parte dell'attributo sensibile *Italian Residence.*

Tuttavia, la maggior parte dei modelli mostra attributi ad essa correlati, come *Distance (km), Residence Country, Residence Italian Region, Residence Italian Province.*

Non è possibile però condannare l'utilizzo della distanza dalla sede lavorativa poiché è spesso essenziale per la scelta di un candidato. Anche in questo caso i modelli sono altamente influenzati dai campi relativi alla RAL.

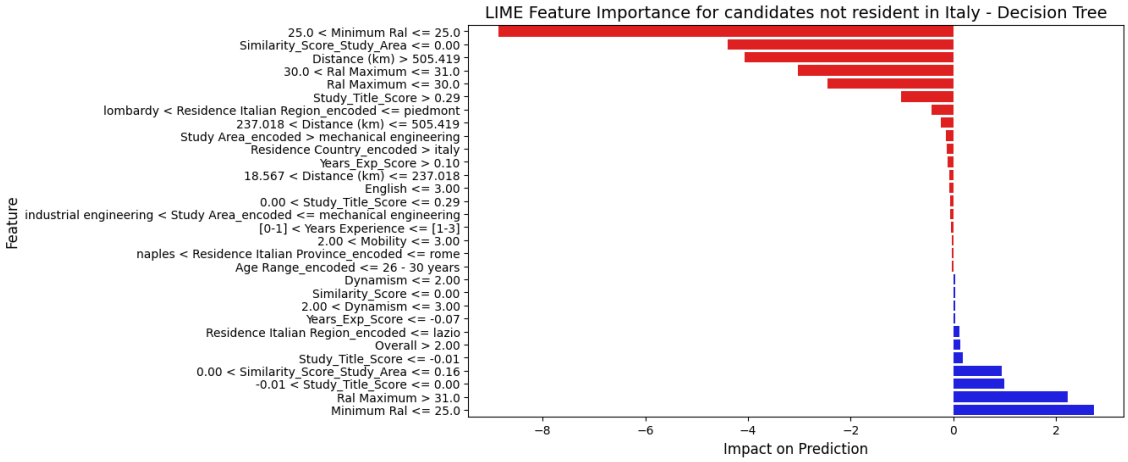

Figura 41 Distribuzione influenza delle 20 feature più importanti per il modello Decision Tree, per candidati con Italian Residence = 'no'

European Residence

Tra i modelli implementati solo `Decision Tree` evidenzia l'utilizzo dell'attributo sensibile *European Residence*, con influenza negativa, come mostrato in Figura 42.

Anche in questo caso tutti i modelli risultano influenzati dai campi correlati, come *Distance (km)* e da quelli relativi alla *RAL*.

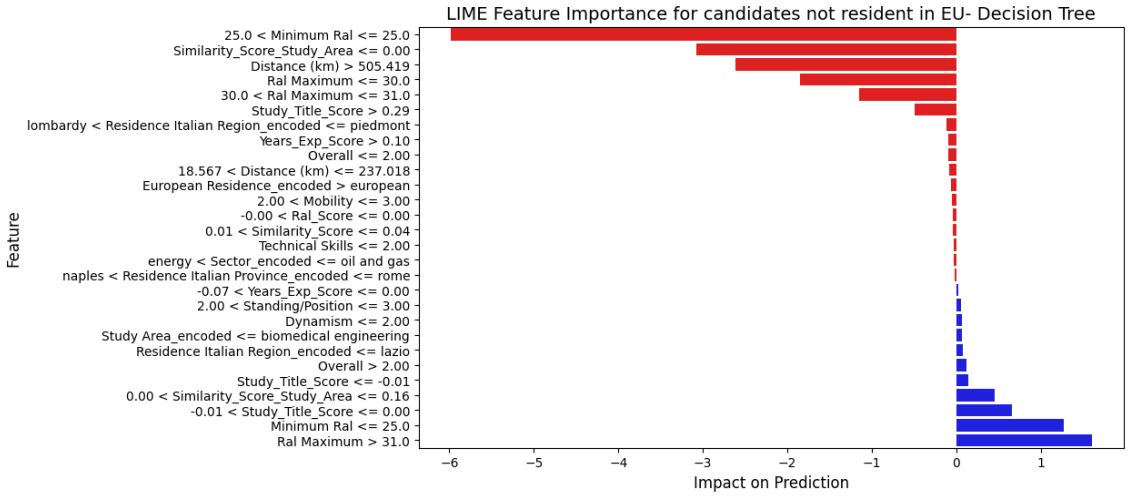

Figura 42 Distribuzione influenza delle 20 feature più importanti per il modello Decision Tree, per candidati con European Residence = 'non-european'

5.4.2. SHAP

L'*explainer* SHAP attribuisce ad ogni feature il cambiamento nella previsione del modello quando condizionata, rivelando così la misura in cui ciascuna caratteristica contribuisce alla classificazione, sia positivamente che negativamente.[19]

Anche con SHAP emergono tra gli attributi più rilevanti quelli legati alla RAL.

Analizzando i risultati della `Neural Network 2`, che aveva ottenuto risultati discutibili per la *Counterfactual Fairness* (Figura 38), emerge l'utilizzo di diversi attributi sensibili. In particolare, analizzando la Figura 43, è possibile notare tra gli attributi:

- *Age Range:* Valori alti di *Age Range* influenzano positivamente il risultato. Anche l'attributo correlato *Years Experience* segue lo stesso andamento.
- *Italian Residence:* Valori bassi della corrispettiva colonna *Italian Residence_encoded* influenzano positivamente la predizione. Questa colonna è binaria e 0 identifica candidati con residenza non italiana. Questo risultato è coerente con i risultati ottenuti dalle metriche.

[19] Goethals, Martens, e Calders, «PreCoF».

- *European Residence:* Per *European Residence* è evidente l'impatto negativo di valori alti, che identificano candidati non europei.

- *Protected Category*: Anche in questo caso *Protected Category_encoded = 1,* che identifica candidati appartenenti a categoria protetta, influenza negativamente il risultato.

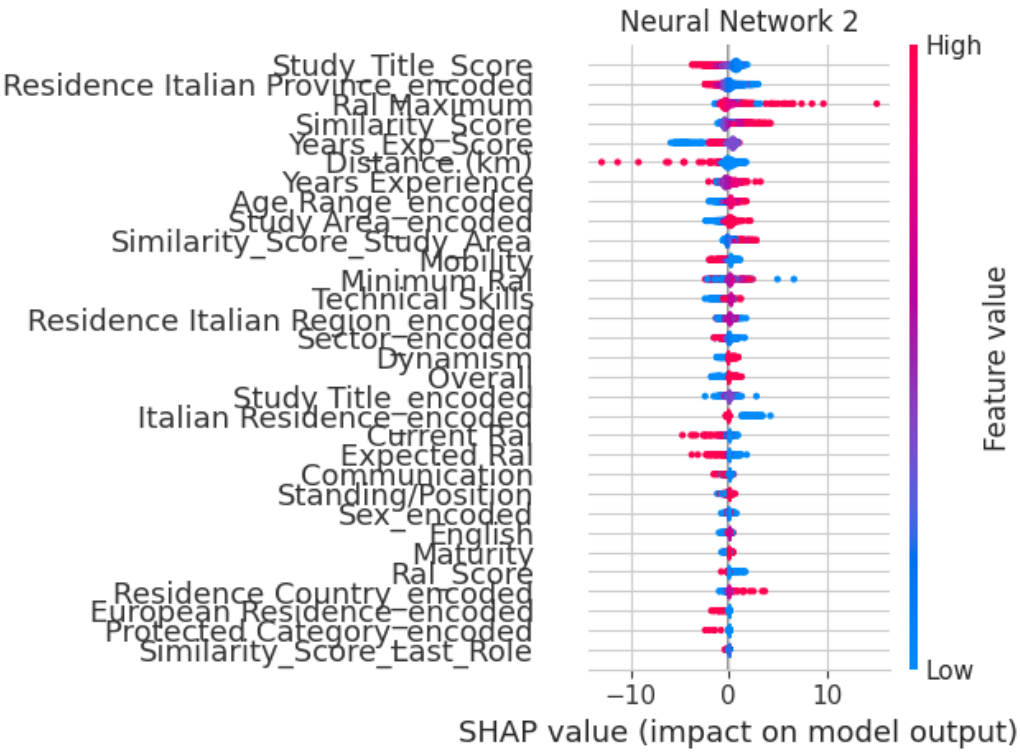

Figura 43 SHAP explanations per la Neural Network 2

6. Mitigazione dei Bias

6.1. Mitigazione pre-processing

L'analisi del dataset non ha evidenziato bias sistematici nei confronti delle categorie protette. Come discusso nella Sezione 4.2.3, nel caso dell'attributo *Sex* le donne presentano un tasso di assunzione leggermente più alto rispetto agli uomini. Questo lieve sbilanciamento potrebbe essere dovuto ad una maggiore competenza delle candidate (Figura 11), sebbene non vi siano dati certi per confermarlo. I recruiter di Akkodis hanno inoltre dichiarato che non esiste all'interno della azienda un processo di compensazione nei confronti delle candidate.

Nel caso degli altri attributi sensibili, come *Protected Category*, *Age Range* o *Italian Residence,* il lieve squilibrio nel tasso di candidati assunti è dovuto principalmente a:

- **Sottorappresentazione** di alcuni gruppi nel dataset;
- **Fattori ragionevoli**, come la distanza dalla sede lavorativa o la mancanza di anni di esperienza necessari.

Per favorire una migliore valutazione delle metriche di *fairness* il *dataset* è stato suddiviso affidando al *test set* una maggiore porzione rispetto alla fase precedente, pari al 40% del totale.

Successivamente sono stati rimossi tutti gli attributi che hanno evidenziato bias durante l'analisi del dataset.

Infine, sono state impiegate tecniche di *oversampling*, tramite `CTGAN`, e *undersampling*, per bilanciare le classi sottorappresentate nel *training set*.

6.1.1. Rimozione Attributi Biased

Nonostante il dataset fornito abbia dimostrato una maggiore equità rispetto ad altri tradizionalmente utilizzati in questo contesto, i campi relativi alla *RAL* hanno evidenziato criticità nascoste.

Come osservato nella Sezione 4.2.3, la RAL prevista dalla posizione lavorativa associata a ciascun candidato assunto sembra essere influenzata dagli attributi sensibili. A parità di titolo di studio o di anni di esperienza è infatti presente una disparità tra candidati appartenenti a gruppi diversi.

Per queste considerazioni tutti i campi relativi a *RAL* sono stati rimossi, per evitare una possibile discriminazione indiretta basata su questi ultimi.

6.1.2. Data Augmentation

Il *training set* è stato ampliato, per compensare la sua dimensione ridotta risultante della diversa suddivisione del dataset, e bilanciato attraverso campioni sintetici, generati tramite **CTGAN** (*Conditional Tabular GAN*), un modello basato su reti generative avversarie progettato per dati tabulari.[20] `CTGAN` è stato scelto per la sua capacità di rappresentazione rispetto ad altri strumenti e per il supporto nativo alle variabili categoriche.

La strategia adottata prevede l'addestramento di un modello `CTGAN` separato per ciascun sottogruppo, identificato da un attributo sensibile e dalla colonna target *Hired.* Questo ha permesso di mantenere le relazioni interne ai sottogruppi, evitando distorsioni nei dati sintetici.

Per ogni modello `CTGAN` addestrato, sono stati generati campioni sintetici fino a raggiungere almeno il 40% della dimensione della classe maggioritaria, relativa allo stesso attributo sensibile, garantendo una presenza più equilibrata per ciascun sottogruppo.

Al termine tutti i campioni sono stati concatenati al *training set* originale, portando il totale dei candidati oltre 24.000.

Dopo l'oversampling sono stati addestrati nuovi modelli, mantenendo gli stessi iperparametri precedentemente ottimizzati con la *Grid Search*.

[20] Panagiotou, Roy, e Ntoutsi, «Synthetic Tabular Data Generation for Class Imbalance and Fairness».

Risultati

Come previsto, l'oversampling ha avuto un impatto mediamente negativo sulle performance dei modelli (Figura 44), ad esclusione di `K Nearest Neighbor`.

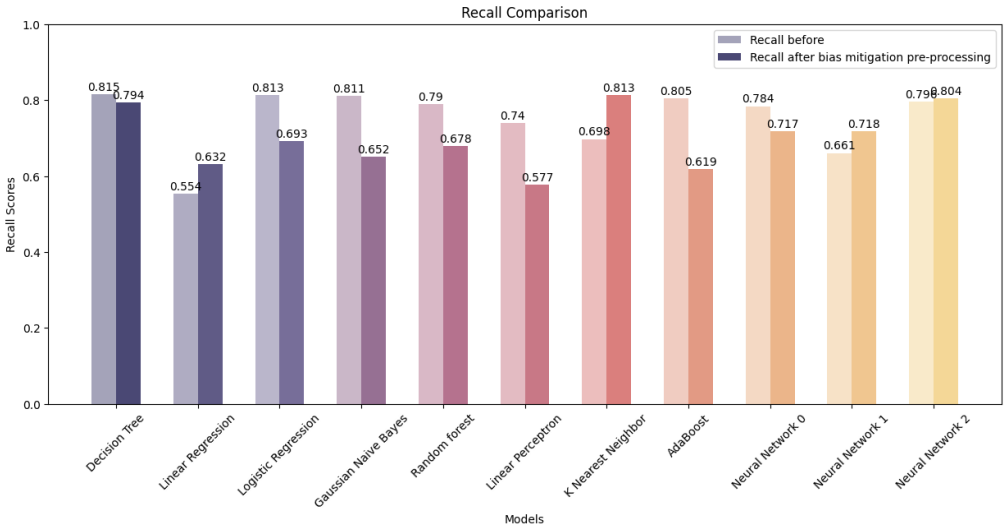

Figura 44 Confronto Recall Score prima e dopo la rimozione delle colonne RAL e l'oversampling con CTGAN

Tuttavia, non ha portato ad un miglioramento evidente della *fairness*. Questo risultato è dovuto principalmente alle dimensioni molto ridotte del *dataset* iniziale. Nonostante il *test set* rappresenti ora il 40% dei candidati, i dati risultano comunque insufficienti per permettere un miglioramento stabile delle metriche, che oscillano invece tra due estremi.

La sottorappresentazione delle categorie protette nel *test set* amplifica l'impatto di qualsiasi variazione nelle predizioni, causando oscillazioni estreme nella direzione del bias.

Nel caso di **Demographic Parity**, in Figura 45, è possibile osservare un cambiamento netto nella direzione dei bias, con modelli che sembrano favorire candidati appartenenti a categoria protetta. Tuttavia, questo apparente sbilanciamento è dovuto al numero limitato di candidati di questo tipo all'interno del test set. Il tasso di predizioni positive cresce infatti molto velocemente per questi candidati, amplificando l'effetto delle predizioni positive aggiuntive.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		< 20 years			
Linear Regression	male				
Logistic Regression	male	< 20 years	yes		
Gaussian Naive Bayes	male				no
Random forest					
Linear Perceptron					
K Nearest Neighbor		< 20 years			
AdaBoost					
Neural Network 0		< 20 years			no
Neural Network 1		< 20 years			
Neural Network 2	male	< 20 years	no	non-european	

Figura 45 Demographic Parity dei modelli addestrati sul training set oversampled, senza campi relativi a RAL

Anche nel caso di **Equalized Odds** si osserva un miglioramento: alcuni bias sono stati risolti mentre altri sono stati migliorati se non addirittura invertiti.

Per il modello `Gaussian Naïve Bayes` è possibile notare che i candidati appartenenti a categoria protetta hanno avuto un tasso di falsi positivi maggiore, evidenziando un cambiamento del modello in questa direzione. Anche in questo caso, il tasso di falsi positivi aumenta rapidamente, poiché i candidati non assunti appartenenti a categoria protetta sono solo 18.

Nel caso di *Italian Residence* è possibile osservare che la maggior parte dei modelli ha risolto il bias mentre i restanti lo hanno invertito nei confronti della classe maggioritaria (*Italian Residence = yes*), evidenziando anche in questo caso un miglioramento, seppur brusco, in questa direzione.

Il cambiamento nei confronti dell'attributo *Sex* è invece da imputare alla rimozione delle colonne riguardanti la *RAL*, che presentavano un forte squilibrio verso le donne. Rimuovendo gli attributi evidentemente discriminanti nei confronti delle candidate e tenendo presente che il dataset di partenza prevedeva già un tasso maggiore di assunzione per le donne, le metriche evidenziano un cambiamento in direzione opposta.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		20 - 25 years [TPR=0.444]		non-european [TPR=0.0]	yes [TPR=0.0]
Linear Regression	male [TPR=0.218]	> 45 years [TPR=0.192]			
Logistic Regression	male [TPR=0.489/FPR=0.148]	36 - 40 years [TPR=0.412]			no [TPR=0.607]
Gaussian Naive Bayes	male [TPR=0.286]	> 45 years [TPR=0.192]			no [TPR=0.362/FPR=0.063]
Random forest					yes [TPR=0.0]
Linear Perceptron		36 - 40 years [TPR=0.059]		european [TPR=0.189]	no [TPR=0.189]
K Nearest Neighbor			yes [TPR=0.644]	european [TPR=0.648]	yes [TPR=0.0]
AdaBoost			no [TPR=0.0]	non-european [TPR=0.0]	no [TPR=0.255]
Neural Network 0	male [TPR=0.391]	< 20 years [TPR=0.353]	yes [TPR=0.454]		
Neural Network 1		< 20 years [TPR=0.294]	yes [TPR=0.459]		
Neural Network 2		< 20 years [TPR=0.412]	no [TPR=0.5]	non-european [TPR=0.0]	yes [TPR=0.5]

Figura 46 Equalized Odds dei modelli addestrati sul training set oversampled, senza campi relativi a RAL

Dall'analisi di **Counterfactual Fairness** emerge chiaramente la nuova direzione dei modelli, che hanno risolto o invertito la direzione del bias.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree					
Linear Regression	male [-2.34%], female [10.45%]				
Logistic Regression	male [-8.49%], female [28.75%]	< 20 years [-4.45%], > 45 years [7.92%]			yes [-10.77%], no [0.06%]
Gaussian Naive Bayes	male [-2.31%], female [8.81%]				no [-0.12%], yes [25.67%]
Random forest					
Linear Perceptron	male [-1.81%], female [8.31%]				
K Nearest Neighbor		< 20 years [0.42%], > 45 years [17.83%]			
AdaBoost					
Neural Network 0		< 20 years [-0.83%], > 45 years [9.38%]	yes [0.68%], no [29.64%]	european [0.18%], non-european [36.02%]	
Neural Network 1		< 20 years [-2.37%], > 45 years [10.12%]	yes [0.5%], no [22.26%]	european [0.15%], non-european [27.18%]	
Neural Network 2		< 20 years [-5.1%], > 45 years [21.69%]	yes [0.71%], no [29.64%]	european [0.21%], non-european [37.72%]	yes [-16.2%], no [0.09%]

Figura 47 Counterfactual Fairness dei modelli addestrati sul training set oversampled, senza campi relativi a RAL

6.1.3. Bilanciamento Dataset

Dopo l'*oversampling* è stato effettuato un ulteriore bilanciamento del *dataset* attraverso undersampling, per equilibrare maggiormente e mitigare eventuali distorsioni introdotte. Questa tecnica di sotto campionamento controllato permette di limitare l'impatto delle classi maggiormente rappresentate.

Il metodo impiegato assegna a ciascun campione un peso inversamente proporzionale alla dimensione del gruppo di appartenenza, permettendo un campionamento bilanciato, senza alterare la dimensione complessiva del *training set*. Durante la selezione sono stati favoriti candidati reali.

Al termine il nuovo *training set* è stato utilizzato per addestrare un nuovo set di modelli.

Risultati

Come atteso, l'undersampling ha avuto un ulteriore impatto negativo sulle performance dei modelli, come mostrato in Figura 48, in cui sono stati messi a confronto i punteggi di recall ottenuti dai modelli nei tre casi visti.

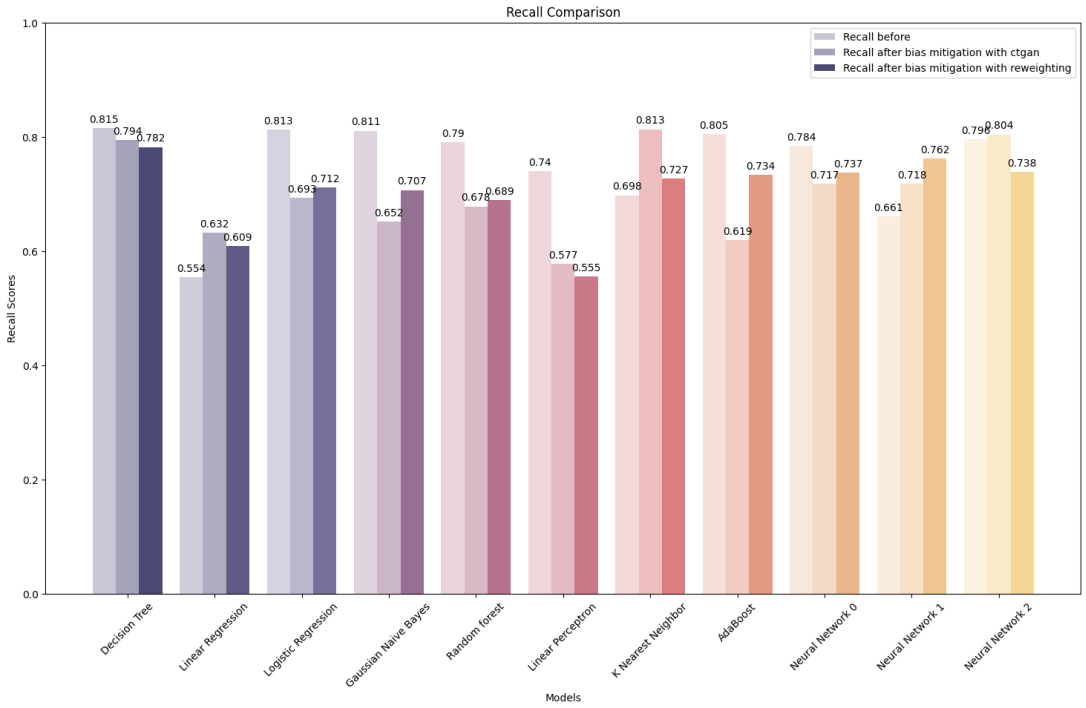

Figura 48 Confronto Recall Score prima e dopo la rimozione delle colonne RAL e l'oversampling con CTGAN

Tuttavia, rispetto alla fase precedente, non si osservano particolari variazioni positive nelle metriche di *fairness*, che evidenziano solo lievi cambiamenti.

Per questo motivo, nella fase successiva, verrà utilizzato il *training set* sovra campionato tramite `CTGAN`.

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		< 20 years			
Linear Regression					
Logistic Regression	male	< 20 years	yes		yes
Gaussian Naive Bayes		< 20 years	yes	european	yes
Random forest		< 20 years			
Linear Perceptron		< 20 years			
K Nearest Neighbor		20 - 25 years			
AdaBoost		< 20 years			
Neural Network 0		< 20 years			
Neural Network 1		< 20 years			
Neural Network 2		< 20 years			

Figura 49 Demographic Parity dei modelli addestrati sul training set undersampled

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree		40 - 45 years [TPR=0.462]			yes [TPR=0.0]
Linear Regression		> 45 years [TPR=0.077]		european [TPR=0.24]	yes [TPR=0.0]
Logistic Regression	male [TPR=0.624/FPR=0.234]	> 45 years [TPR=0.538], < 20 years [FPR=0.233]		non-european [TPR=0.5]	yes [TPR=0.0/FPR=0.056]
Gaussian Naive Bayes		> 45 years [TPR=0.385]	yes [FPR=0.15]	european [TPR=0.571/FPR=0.152]	yes [TPR=0.0]
Random forest		26 - 30 years [TPR=0.275]			yes [TPR=0.0]
Linear Perceptron		> 45 years [TPR=0.038]		european [TPR=0.143]	
K Nearest Neighbor		26 - 30 years [TPR=0.375]	yes [TPR=0.495]		yes [TPR=0.0]
AdaBoost		> 45 years [TPR=0.346]	no [TPR=0.0]	non-european [TPR=0.0]	yes [TPR=0.0]
Neural Network 0		< 20 years [TPR=0.294]	yes [TPR=0.495]		
Neural Network 1		< 20 years [TPR=0.294]			
Neural Network 2		26 - 30 years [TPR=0.362]	yes [TPR=0.5]		

Figura 50 Equalized Odds dei modelli addestrati sul training set undersampled

	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Decision Tree					
Linear Regression					
Logistic Regression	male [-5.04%], female [17.66%]	< 20 years [-5.85%], > 45 years [9.26%]			yes [-28.66%], no [0.27%]
Gaussian Naive Bayes					yes [-18.61%], no [0.21%]
Random forest					
Linear Perceptron					
K Nearest Neighbor		< 20 years [-0.12%], > 45 years [13.09%]			
AdaBoost					
Neural Network 0		< 20 years [-1.34%], > 45 years [13.26%]	yes [0.77%], no [23.83%]	european [0.0%], non-european [33.77%]	
Neural Network 1		< 20 years [-1.93%], > 45 years [21.66%]	yes [0.98%], no [39.23%]	european [0.18%], non-european [43.65%]	
Neural Network 2		< 20 years [-1.42%], > 45 years [11.36%]	yes [0.77%], no [19.53%]	european [0.06%], non-european [35.7%]	

Figura 51 Counterfactual Fairness dei modelli addestrati sul training set undersampled

6.2. Mitigazione in-processing: Adversarial Debiasing

L'*Adversarial Debiasing* è una tecnica basata su reti neurali avversarie, utilizzata per mitigare i *bias* di un modello durante l'addestramento. L'implementazione prevede l'affiancamento al modello principale (*Main Model*) di modelli avversari (*Adversary Models*), che hanno come obiettivo la predizione di uno specifico attributo sensibile, utilizzando come input le predizioni del *Main Model*. Se le reti avversarie effettuano predizioni accurate significa che il modello principale è particolarmente influenzato dagli attributi sensibili nelle sue predizioni.

L'addestramento avviene in modo alternato tra il modello principale e i modelli avversari. In questo caso la rete neurale non si limita a minimizzare la propria funzione di perdita, ma ottimizza una *loss* combinata, che include la differenza tra la sua funzione di perdita e quella dei modelli avversari. Anche l'aggiornamento dei pesi del modello principale è diretto dalla funzione di perdita combinata. Questo spinge il modello principale a trovare rappresentazioni che siano predittive per il suo target ma non per gli attributi sensibili, riducendo così i *bias*.

Se i modelli avversari hanno accuratezza elevata, significa che il *Main Model* sta ancora sfruttando gli attributi sensibili nelle sue decisioni. Viceversa, se la *loss* degli avversari aumenta il modello principale ha appreso una rappresentazione più equa.

6.2.1. Architettura

- ***Main Model***: IL *Main Model* ha la stessa struttura delle reti neurali finora impiegate, composta da tre *hidden layer*, *batch normalization* e *dropout.* I parametri utilizzati coincidono con quelli ottimizzati tramite *Grid Search* nelle fasi precedenti.

- ***Adversary Models***: La struttura delle reti avversarie è più semplice ed è costituita da due *hidden layer* e *batch normalization.* A seconda dell'attributo sensibile associato l'*output layer* è a singolo neurone, per *feature* binarie come *Sex,* o a neuroni multipli con attivazione *softmax.* La funzione di perdita di ciascun modello avversario è pesata in base alla distribuzione delle classi dell'attributo sensibile, per evitare che la classe maggioritaria domini l'addestramento.

6.2.2. Equalized Odds Penalty

È stata inoltre implementata una funzione di penalità basata su *Equalized Odds*, integrata nella funzione di perdita complessiva. Quest'ultima misura le differenze nei tassi di errore tra gruppi sensibili e spinge il modello ad effettuare predizioni più eque tra i diversi sottogruppi.

```
def equalized_odds_penalty(y_true, y_pred, sensitive_group):

    for group in groups:
        tn, fp, fn, tp = confusion_matrix(group['true_labels'], group['predictions'], labels=[0, 1])
        tpr = tp / (tp + fn) if tp + fn != 0 else 0
        fpr = fp / (fp + tn) if fp + tn != 0 else 0

    max_tpr_diff = max(tprs) - min(tprs)
    max_fpr_diff = max(fprs) - min(fprs)

    penalty = max(max_tpr_diff, max_fpr_diff)
    return penalty
```

Figura 52 Pseudo codice per Equalized Odds Penalty

6.2.3. Addestramento

L'addestramento è stato effettuato utilizzando il *training set* ottenuto tramite *oversampling* con `CTGAN`, per combinare tecniche di *bias mitigation* diverse.

Il *Main Model* è stato addestrato sequenzialmente con ognuna delle cinque reti avversarie, relative agli attributi sensibili: *Sex, Age Range, Italian Residence, European Residence* e *Protected Category.*

Risultati

Le performance del modello si presentano inferiori ai modelli non sottoposti a tecniche di *bias mitigation*. Analizzando i punteggi ottenuti dalla rete principale emerge la sua tendenza nell'etichettare negativamente buona parte dei campioni.

Questo comportamento più cauto è dovuto alla competizione con le reti avversarie. Essendo il *test set* sbilanciato nei confronti dell'etichetta positiva, la *accuracy* del modello si mantiene comunque alta.

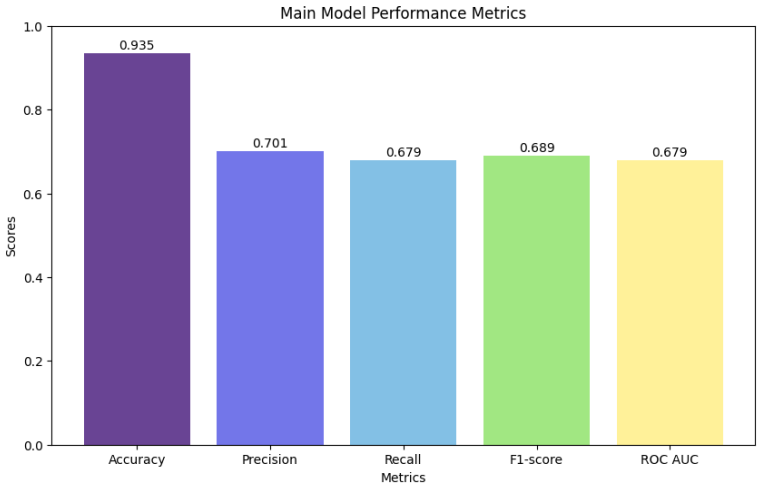

Figura 53 Performance Main Model Adversarial Debiasing

Tuttavia, le metriche di fairness mostrano miglioramenti promettenti. Anche in questo caso *Counterfactual Fairness*, in Figura 54, evidenzia un cambiamento di rotta da parte del modello, anche se da correggere.

Metric	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Demographic Parity		< 20 years			
Equalized Odds		< 20 years [TPR=0.118]		non-european [TPR=0.0]	
Counterfactual Fairness	male [-0.15%], female [0.92%]	< 20 years [-1.66%], > 45 years [15.1%]	yes [0.5%], no [20.42%]	european [0.09%], non-european [28.96%]	

Figura 54 Metriche ottenute per il Main Model dopo l'Adversarial Debiasing

Explainability

Il grafico ottenuto con SHAP evidenzia una forte dipendenza nei confronti di provincia (*Residence Italian Province*) e regione di residenza (*Residence Italian Region*). Questo risultato è dovuto sia alla bassa dimensionalità del *test set*, che risulta poco rappresentativo, sia al fatto che la maggior parte delle posizioni lavorative gestite sono concentrate in città specifiche, come Milano, Torino, etc (Figura 56).

È abbastanza ragionevole che candidati residenti nella stessa città siano avvantaggiati.

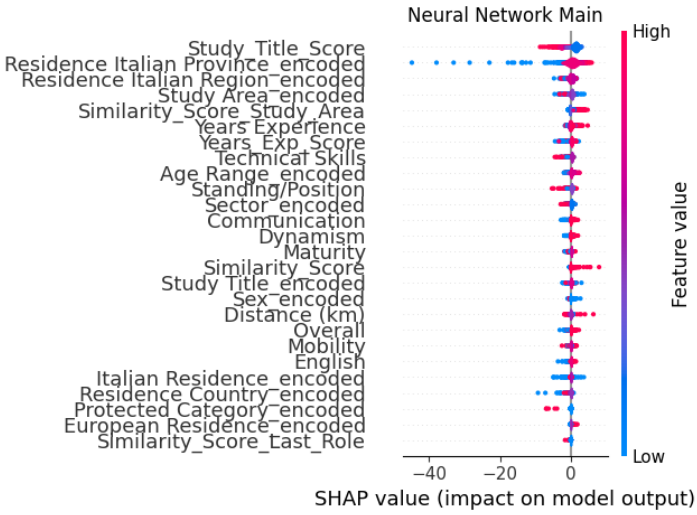

Figura 55 Shap Explainer per il Main Model

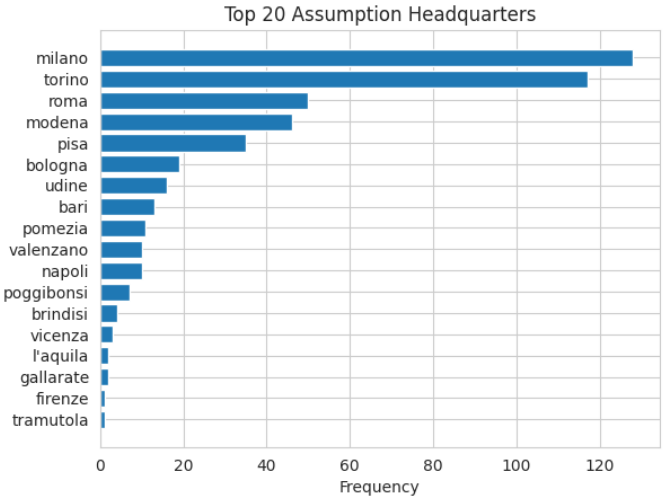

Figura 56 Distribuzione delle 20 sedi lavorative più frequenti presenti nel dataset

6.2.4. Selezione Soglia Ottimale

Per migliorare le performance complessive è stata calcolata una soglia ottimale, attraverso la *ROC Curve* (Figura 57), per calcolare il punto di maggior bilanciamento tra TPR e FPR.

Utilizzando la soglia ottenuta in questa fase le performance complessive del modello migliorano, mentre la sua accuratezza peggiora, poiché classifica positivamente più candidati.

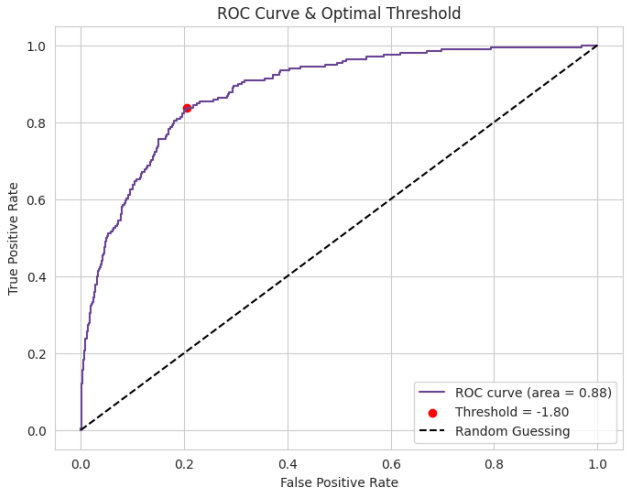

Figura 57 Curva ROC per Main Model

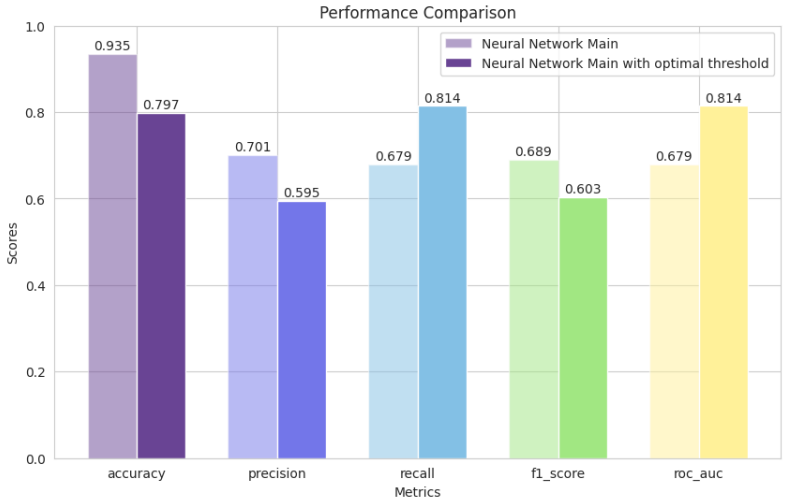

Figura 58 Confronto delle performance del Main Model prima e dopo l'utilizzo della soglia ottima

Tuttavia, l'utilizzo di una soglia unica può non essere la scelta migliore a livello di *fairness*. Le nuove metriche calcolate dopo l'applicazione della soglia sono rappresentate in Figura 59. I risultati evidenziano che il modello favorisce ancora alcune categorie.

	Metric	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Neural Network Main	Demographic Parity	male	20 - 25 years	no	non-european	
Neural Network Main	Equalized Odds		< 20 years [TPR=0.647]		non-european [TPR=0.5]	yes [TPR=0.5]
Neural Network Main	Counterfactual Fairness	male [-19.17%], female [-18.1%]	< 20 years [-20.68%], > 45 years [-3.92%]	yes [-18.52%], no [1.39%]	european [-18.93%], non-european [9.94%]	yes [-24.07%], no [-19.08%]

Figura 59 Metriche Main Model con soglia ottimale

6.3. Mitigazione post-processing: Fair Thresholding

Nei modelli di *machine learning*, l'utilizzo di una soglia fissa per la classificazione può introdurre *bias*, penalizzando alcuni gruppi più di altri. Nella fase precedente l'utilizzo della soglia ottenuta tramite *ROC Curve* ha portato ad una disparità nelle prestazioni del modello nei confronti di gruppi diversi, come mostrato dalle metriche in Figura 59.

Per risolvere questo sbilanciamento, in post-processing, sono state calcolate soglie differenziate per ciascun sottogruppo, ottimizzate in base alla rispettiva *ROC Curve*. Questo ha permesso di ridurre gli errori di classificazione a livello locale, migliorando l'equilibrio tra sensibilità e specificità del modello per ogni gruppo.

L'utilizzo di soglie personalizzate mostra un miglioramento delle performance, come mostrato in Figura 60.

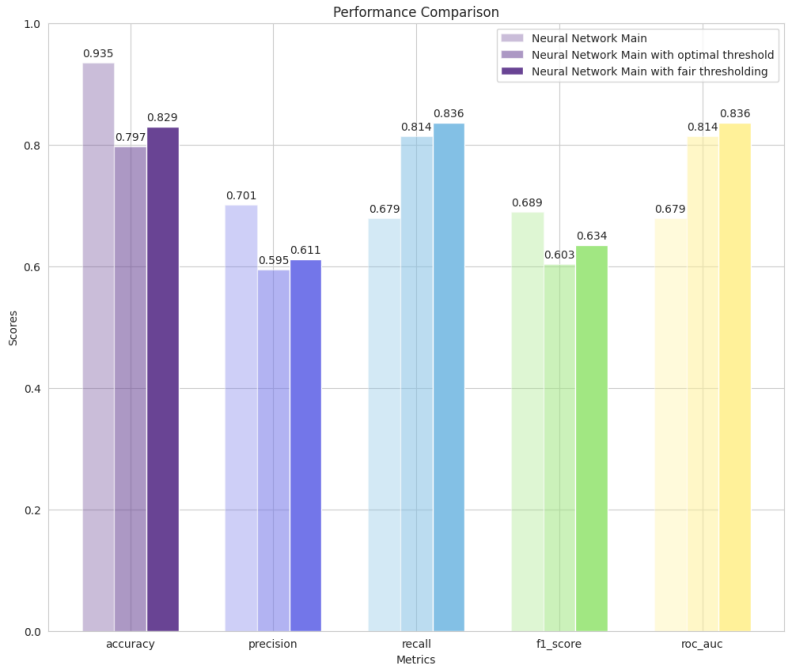

Figura 60 Confronto performance del Main Model a seconda del tipo di soglia utilizzata

Le metriche di *fairness* ottenute dopo l'applicazione delle soglie personalizzate risultano migliorate, come illustrato in Figura 61.

Per **Demographic Parity** la rete non discrimina più i candidati appartenenti alla categoria *male* mentre le predizioni nel caso dell'attributo *Age Range* risultano ora sbilanciate nei confronti della fascia *36 – 40 years*, che rappresenta il gruppo con il maggiore tasso di assunzione nel dataset (Figura 18).

Equalized Odds non evidenzia più criticità grazie all'utilizzo di soglie ottimizzate con la ROC curve.

Nel caso di **Counterfactual Fairness** si osserva un miglioramento, con una riduzione della disparità tra le classi di uno stesso attributo sensibile.

	Metric	Sex_encoded	Age Range_encoded	Italian Residence_encoded	European Residence_encoded	Protected Category_encoded
Neural Network Main	Demographic Parity		36 - 40 years	no	non-european	
Neural Network Main	Equalized Odds					
Neural Network Main	Counterfactual Fairness	male [-16.08%], female [-15.01%]	< 20 years [-17.6%], > 45 years [-0.83%]	yes [-15.43%], no [4.48%]	european [-15.85%], non-european [13.03%]	yes [-20.98%], no [-15.99%]

Figura 61 Metriche di fairness dopo l'applicazione di soglie diverse per ciascun sottogruppo

7. Discussione

L'analisi condotta sul dataset fornito da Akkodis ha evidenziato quanto sia importante considerare la *fairness* nello sviluppo di modelli di *machine learning*, soprattutto in contesti critici come quello trattato dalla azienda. L'equità non rappresenta solo una questione etica, ma favorisce decisioni affidabili e prive di distorsioni sistematiche.

Questo studio si è concentrato sull'identificazione e la valutazione dei *bias* presenti nel *dataset* e successivamente acquisiti dai modelli con esso sviluppati. L'identificazione è avvenuta attraverso metriche selezionate, in grado di fornire una visione dell'equità da prospettive diverse. Il livello di *fairness* di un modello non può essere valutato con una singola metrica ma richiede una analisi approfondita e multidisciplinare, valutando oltre alle prestazioni complessive anche la distribuzione degli errori nei singoli gruppi.

Per mitigare le disparità riscontrate sono state impiegate diverse strategie di mitigazione lungo tutto il processo di sviluppo del modello. In fase di *pre-processing*, il dataset è stato bilanciato per ridurre i *bias* più evidenti e compensare la sottorappresentazione di alcune classi. Durante l'addestramento del modello è stata inoltre utilizzata la tecnica di *Adversarial Debiasing*, volta a ridurre l'utilizzo di attributi sensibili da parte della rete. Infine, in *post-processing*, sono state elaborate soglie *ad hoc* per ciascun sottogruppo del dataset, ottimizzate tramite le *ROC Curve*, per garantire buone prestazioni a livello locale.

Il livello di equità è stato monitorato durante tutto lo sviluppo, per valutare *real-time* l'impatto di ciascuna tecnica applicata. Tuttavia, il limite principale di questa tesi è dato dalla dimensione ridotta del dataset, che ha reso più difficile evidenziare il pieno potenziale delle tecniche di mitigazione applicate, oltre che dalla struttura originale che presentava posizioni lavorative solo per candidati assunti. Dataset più ampi oltre a migliorare le performance generali di un modello consentono anche una valutazione più robusta dell'equità.

Dai risultati ottenuti emerge con chiarezza la necessità di bilanciare prestazioni e *fairness*. Modelli che mostrano prestazioni molto alte, quando testati su dati discriminanti, non sono necessariamente i migliori, l'accuratezza evidenzia infatti la loro capacità nell'emulare le discriminazioni umane.

Viceversa, una leggera riduzione nelle prestazioni non comporta necessariamente un modello meno efficace, ma potrebbe indicare che l'algoritmo sta selezionando i candidati esclusivamente in base alle loro competenze, senza essere influenzato dai bias umani presenti nel test set. Modelli equi possono apparire meno performanti rispetto a modelli biased, poiché generano previsioni diverse da quelle attese, sfidando le dinamiche discriminatorie preesistenti.

Tuttavia, la vera svolta non è mitigare le disparità nei modelli, che risultano inevitabilmente meno performanti se valutati su *test set biased*, ma dare il "buon esempio", generando dataset più bilanciati e rappresentativi. I modelli di AI non sono solo in grado di velocizzare i processi decisionali, ma possono contribuire a renderli più equi e inclusivi.

La loro forte influenza sul pensiero umano non dovrebbe essere utilizzata per perpetuare le disparità del passato, ma per garantire un futuro più giusto per tutti.

Bibliografia

- Aasheim, Tor H, e Knut T Hufthammer. «Bias Mitigation with AIF360: A Comparative Study», s.d.

- Adecco Group. «The Adecco Group completa l'acquisizione della quota di maggioranza di AKKA Technologies». *The Adecco Group* (blog), 24 febbraio 2022. https://adeccogroup.it/the-adecco-group-completa-lacquisizione-della-quota-di-maggioranza-di-akka-technologies/.

- Alikhademi, Kiana, Brianna Richardson, Emma Drobina, e Juan E. Gilbert. «Can Explainable AI Explain Unfair ness? A Framework for Evaluating Explainable AI». arXiv, 14 giugno 2021. https://doi.org/10.48550/arXiv.2106.07483.

- Bellamy, Rachel K. E., Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, et al. «AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias». arXiv, 3 ottobre 2018. https://doi.org/10.48550/arXiv.1810.01943.

- Bird, Sarah, Miroslav Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, Kathleen Walker, e Allovus Design. «Fairlearn: A Toolkit for Assessing and Improving Fairness in AI», s.d.

- Diversity, Equity & Inclusion. «Akkodis.com», s.d.

- Fisher Phillips. «New York Lawmakers Aim to Close Loopholes in NYC's AI Bias Audit Law and Add Teeth to Workplace Protections». Consultato 7 marzo 2025. https://www.fisherphillips.com/en/news-insights/new-york-lawmakers-aim-to-close-loopholes-nycs-ai-bias-audit-law.html.

- Gilliland, Stephen W. «The Perceived Fairness of Selection Systems: An Organizational Justice Perspective». *The Academy of Management Review* 18, fasc. 4 (1993): 694–734. https://doi.org/10.2307/258595.

- github. «Countries.csv». Github. Consultato 24 febbraio 2025. https://raw.githubusercontent.com/google/dspl/master/samples/google/canonical/countries.csv.

- Goethals, Sofie, David Martens, e Toon Calders. «PreCoF: Counterfactual Explanations for Fairness». *Machine Learning* 113, fasc. 5 (maggio 2024): 3111–42. https://doi.org/10.1007/s10994-023-06319-8.

- HenryChinaski, Matteo. «Comuni-Italiani-2018-Sql-Json-excel». Github. github. Consultato 24 febbraio 2025. https://github.com/MatteoHenryChinaski/Comuni-Italiani-2018-Sql-Json-excel/tree/master.

- Holzinger, Andreas, Randy Goebel, Ruth Fong, Taesup Moon, Klaus-Robert Müller, e Wojciech Samek, a c. di. *xxAI - Beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020, July 18, 2020, Vienna, Austria, Revised and Extended Papers*. Vol. 13200. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2022. https://doi.org/10.1007/978-3-031-04083-2.

- Madaio, Michael, Lisa Egede, Hariharan Subramonyam, Jennifer Wortman Vaughan, e Hanna Wallach. «Assessing the Fairness of AI Systems: AI Practitioners' Processes, Challenges, and Needs for Support». arXiv, 10 febbraio 2022. https://doi.org/10.48550/arXiv.2112.05675.

- Mehrabi, Ninareh, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, e Aram Galstyan. «A Survey on Bias and Fairness in Machine Learning». arXiv, 25 gennaio 2022. https://doi.org/10.48550/arXiv.1908.09635.

- Mujtaba, Dena F., e Nihar R. Mahapatra. «Fairness in AI-Driven Recruitment: Challenges, Metrics, Methods, and Future Directions». arXiv, 2 giugno 2024. https://doi.org/10.48550/arXiv.2405.19699.

- «Packed with Loopholes: Why the AI Act Fails to Protect Civic Space and the Rule of Law | ECNL», 4 marzo 2024. https://ecnl.org/news/packed-loopholes-why-ai-act-fails-protect-civic-space-and-rule-law.

- Panagiotou, Emmanouil, Arjun Roy, e Eirini Ntoutsi. «Synthetic Tabular Data Generation for Class Imbalance and Fairness: A Comparative Study». arXiv, 8 settembre 2024. https://doi.org/10.48550/arXiv.2409.05215.

- Patalay, Prathamesh. «COMPAS : Unfair Algorithm ?» *Medium* (blog), 22 novembre 2023. https://medium.com/@lamdaa/compas-unfair-algorithm-812702ed6a6a.