

POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Gestionale

Classe L8 – Ingegneria dell'informazione

Anno accademico 2021/2022

Sessione di Laurea Dicembre 2022

TOOL PER ANALISI DEL GENDER GAP E SIMULAZIONE RICERCA DI UN TEAM WORK



**Politecnico
di Torino**

Relatore:

Fulvio Corno

Candidato:

Sylvie Molinatto

Sommario

1. Proposta del progetto	3
1.1 Descrizione del problema proposto	3
1.2 Descrizione della rilevanza gestionale del problema	3
1.3 Descrizione dei data-set per la valutazione	3
1.4 Descrizione preliminare degli algoritmi coinvolti	4
1.5 Descrizione preliminare delle funzionalità previste per l'applicazione software	4
2. Descrizione dettagliata del problema affrontato	4
2.1 Analisi dei dati presenti nel data-set	4
2.2 Ricorsione e ricerca del personal	6
3. Descrizione del data-set utilizzato per l'analisi	7
3.1 Struttura data-set	7
4. Diagramma delle classi principali e descrizione degli algoritmi coinvolti	8
4.1 Strutture dati	8
4.2 Il package 'it.polito.tdp.Tesi'	9
4.3 Il package 'it.polito.tdp.Tesi.db'	11
4.4 Il package 'it.polito.tdp.Tesi.model'	13
5. Esempi di utilizzo dell'applicazione	17
5.1 Esempio analisi dei dati	17
5.2 Esempio ricerca del personale	18
5.3 Link al video YouTube	19
6. Valutazione dei risultati e conclusioni	20
6.1 Valutazione dei risultati	20
6.2 Conclusioni	20

1 - Proposta del progetto

1.1 Descrizione del problema proposto

La candidata propone di realizzare un tool che permetta di visualizzare in maniera semplice e immediata alcuni dati riguardanti la differenza di genere in alcuni settori lavorativi. In particolare, il tool si propone di valutare la percentuale di rappresentanza femminile nei settori considerati (segregazione orizzontale), le differenze salariali di genere e altri fattori presi da un dataset. Inoltre, l'applicazione ha l'ulteriore funzione di permettere all'utente di aprire delle candidature per determinate posizioni lavorative e ottenere il team work che rispetti maggiormente i requisiti richiesti sia liberamente sia con il vincolo di ottenere un team con un'equa rappresentanza di genere.

1.2 Descrizione della rilevanza gestionale del problema

Poiché il tema del divario di genere è piuttosto dibattuto, la realizzazione di un tool che possa analizzare alcuni dati e mostrarne i risultati in maniera chiara e intuitiva risulta essere un modo pratico per diffondere consapevolezza della problematica esistente.

Conseguentemente la possibilità di effettuare la ricerca di un team di lavoro in cui sia massimizzata la professionalità costituisce un'ulteriore fonte di analisi in termini di disparità di genere.

1.3 Descrizione dei data-set per la valutazione

Il data-set utilizzato è tratto da Kaggle al link

<https://www.kaggle.com/datasets/nilimajauhari/glassdoor-analyze-gender-pay-gap>

Si tratta di una raccolta di dati relativi a utenti iscritti al sito web Glassdoor nel quale gli impiegati e gli ex impiegati di un'azienda anonimamente recensiscono le aziende e i loro superiori.

Il dataset contiene i seguenti attributi:

- Id – attributo aggiunto manualmente per distinguere i vari record
- JobTitle (Data Scientist, Driver, Financial Analyst, Graphic Designer, IT, Manager, Marketing Associate, Sales Associate, Software Engineer, Warehouse Associate)
- Gender (Male, Female)
- Age
- PerfEval (Performance Evaluation Score)
- Education (College, High School, Masters, PhD)
- Dept (Department) (Administration, Engineering, Operations, Management, Sales)
- Seniority
- BasePay (Annual Basic Pay in \$)
- Bonus (Annual Bonus Pay in \$)

La dimensione del dataset è di 1000 righe di cui 532 riguardanti utenti di genere maschile e 468 di genere femminile.

1.4. Descrizione preliminare degli algoritmi coinvolti

Poiché il dataset è in formato CSV, come primo passaggio è necessario effettuare una conversione in linguaggio SQL. Al dataset viene aggiunto un attributo, nominato 'id', per distinguere i vari record. L'applicazione viene realizzata in linguaggio Java e segue i pattern MVC (Model View Controller) e DAO (Data Access Object) per ottenere separazione tra interfaccia utente, logica applicativa e accesso ai dati.

Successivamente per effettuare l'analisi relativa alla prima funzione del tool, è necessario creare i metodi che permettano all'utente di formulare le query per la rilevazione delle percentuali riguardanti un determinato settore lavorativo e generare i grafici per la visualizzazione dei risultati. In seguito, per realizzare la funzione di ricerca di un team, si utilizzerà un algoritmo ricorsivo che permetterà di ottenere le specifiche richieste dall'utente tramite interfaccia congiuntamente alla massimizzazione degli altri attributi. Inoltre, al termine della ricorsione verranno visualizzate alcune statistiche.

1.5 Descrizione preliminare delle funzionalità previste per l'applicazione software

L'applicazione prevede che l'utente possa scegliere un determinato settore lavorativo (attributo jobTitle), tramite menù a tendina, e cliccare un bottone 'Analizza' al fine di ottenere la percentuale di occupazione di genere nel settore e altri dati quali lo stipendio medio di genere, il bonus medio, il livello di istruzione e l'età media.

I risultati verranno mostrati in maniera chiara grazie all'utilizzo di grafici a torta e a barre.

Successivamente l'utente potrà decidere di aprire la candidatura a diverse posizioni lavorative per creare un team di lavoro e, per mezzo di un algoritmo ricorsivo, verranno costituiti due team di lavoro, il primo in cui si cercherà unicamente di massimizzare la somma degli attributi 'Performance Score Evaluation', il secondo in cui si cercherà di realizzare anche l'equità di genere in termini di rappresentanza. A seguito della creazione dei team si potrà visionare la percentuale di genere e le differenze in termini di 'Performance Evaluation Score' medio.

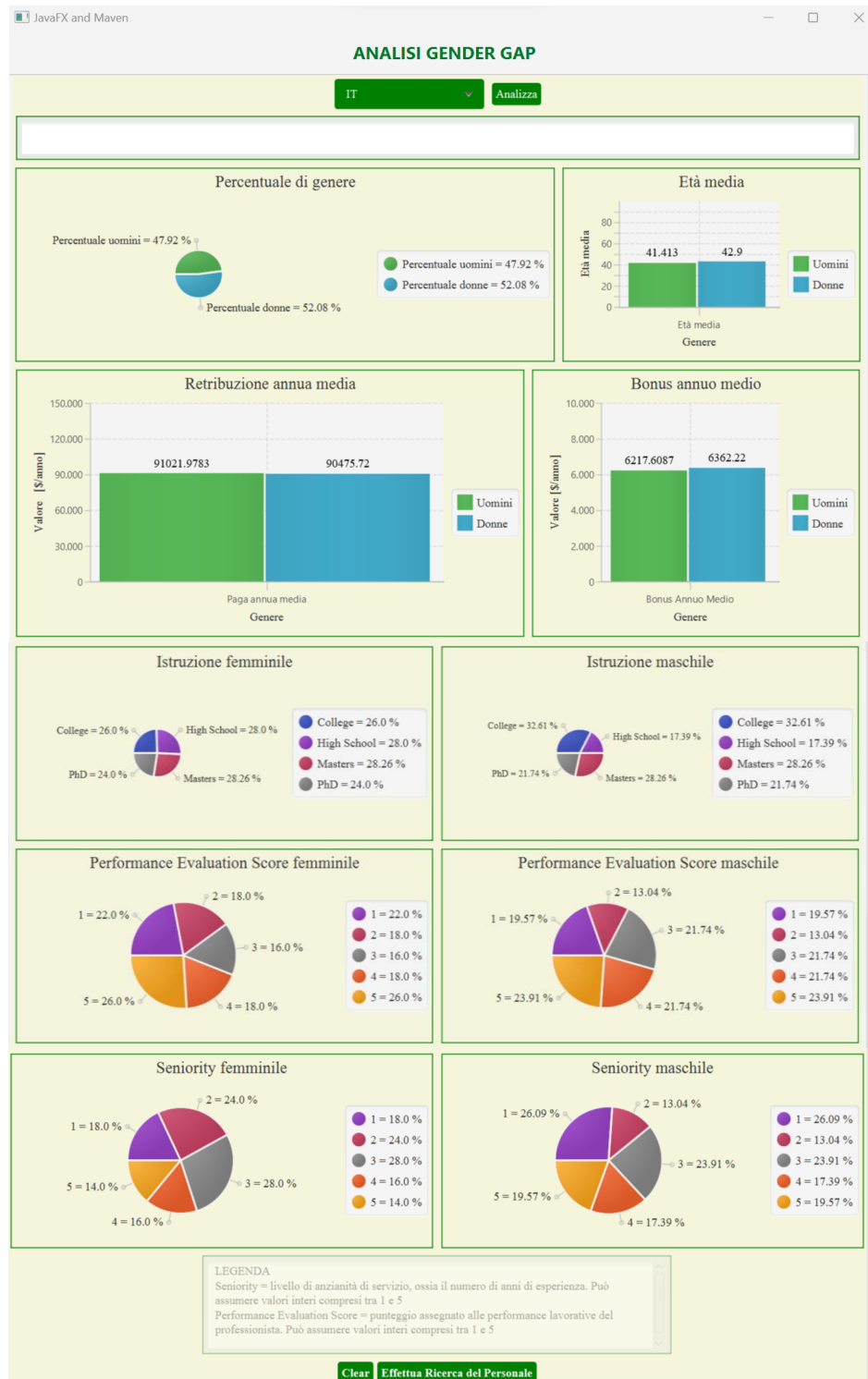
2 - Descrizione dettagliata del problema affrontato

2.1 Analisi dei dati presenti nel dataset

L'obiettivo della realizzazione dell'applicazione è quello di permettere una chiara lettura dei dati riguardanti il divario di genere per mezzo dell'utilizzo di grafici e strumenti che possano riassumere l'informazione contenuta all'interno del dataset in maniera significativa e facilmente comprensibile. L'interfaccia grafica consente all'utente di fruire in maniera più agevole delle informazioni celate all'interno di un dataset che, per sua natura, risulta di difficile lettura.

Per realizzare la funzione di analisi dei dati è stato necessario realizzare una serie di query in linguaggio SQL (Structured Query Language) al database che permettessero di estrapolare il

contenuto informativo di maggiore interesse per l'utente. In particolare, l'analisi è stata condotta a livello di settore lavorativo selezionato dall'utente. Per ciascun settore il programma calcola la percentuale di rappresentanza di genere, l'età media per ciascun genere, le differenze tra gli stipendi e i bonus annuali, la distribuzione dei titoli di studio, dei punteggi assegnati alle performance dei lavoratori e del grado di anzianità di servizio. Successivamente i valori vengono visualizzati tramite grafici a barre e a torta per rendere più facile la comparazione dei risultati.



2.2 Ricorsione e ricerca del personale

Per realizzare la funzione di ricerca del personale idoneo ad un team work si è scelto di permettere all'utente di scegliere, per ogni titolo di lavoro, il numero di professionisti richiesti congiuntamente al livello di istruzione e al grado di anzianità di servizio. Una volta selezionati tali dati, sono stati realizzati due algoritmi ricorsivi che permettono di ottenere due soluzioni diverse. Il primo algoritmo ha l'unico obiettivo di massimizzare la somma dei vari 'Performance Evaluation Score' rispettando le specifiche inserite dall'utente mentre il secondo cerca anche di mantenere equa la rappresentanza di genere. Qualora non ci siano abbastanza professionisti per poter realizzare un team work verrà visualizzato un messaggio di errore.

RICERCA PERSONALE

SETTORE

- DATA SCIENTIST
- DRIVER
- FINANCIAL ANALYST
- GRAPHIC DESIGNER
- IT
- MANAGER
- MARKETING ASSOCIATE
- SALES ASSOCIATE
- SOFTWARE ENGINEER
- WAREHOUSE ASSOCIATE

QUANTITÀ

0 1 2 3 4 5

SENIORITY

5

EDUCATION

PhD

Cerca Teamwork

SOLUZIONE LIBERA

ID	JOB TITLE	GENDER	AGE	PERFORM...	EDUCA
8	Software E...	Male	18	4	PhD
35	Data Scien...	Female	45	5	PhD

Rappresentanza di genere

● Percentuale uomini = 83.33 %
● Percentuale donne = 16.67 %

SOLUZIONE EQUIGENERE

ID	JOB TITLE	GENDER	AGE	PERFORM...	EDUCA
8	Software E...	Male	18	4	PhD
35	Data Scien...	Female	45	5	PhD

Rappresentanza di genere

● Percentuale uomini = 50.0 %
● Percentuale donne = 50.0 %

Performance Evaluation Score medio : 4.83

Performance Evaluation Score medio : 4.0

Clear Ritorna all'analisi dei dati

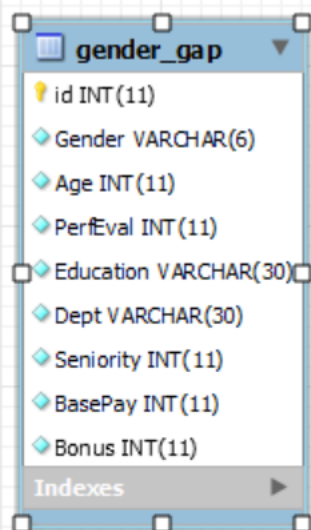
3 - Descrizione del data-set utilizzato per l'analisi

Il database utilizzato per la realizzazione dell'applicazione è reperibile al seguente indirizzo :

<https://www.kaggle.com/datasets/nilimajauhari/glassdoor-analyze-gender-pay-gap>

Si tratta di una raccolta di recensioni anonime di aziende effettuate da dipendenti o ex-dipendenti estrapolata dal sito web Glassdoor. I dati sono stati convertiti dal formato csv a SQL ed è stato aggiunto un attributo, denominato 'id', per distinguere i vari record.

3.1 Struttura data-set



La struttura del database è semplice in quanto vi è un'unica tabella costituita da dieci attributi.

L'attributo 'id' è la chiave primaria, ossia un valore intero che identifica univocamente ciascun record del data-set. Gli attributi 'Gender', 'PerfEval', 'Education', 'Dept' e 'Seniority' sono categorici in quanto possono assumere solo un numero finito di valori e in particolare:

- L'attributo 'Gender' assume unicamente i valori 'Male' e 'Female'.
- L'attributo 'PerfEval' rappresenta il Performance Evaluation Score, ossia il punteggio assegnato alle performance lavorative del professionista, e può assumere solo i valori interi compresi tra 1 e 5.
- L'attributo 'Education' rappresenta il titolo di studio conseguito dal lavoratore e può assumere i valori 'College', 'High School', 'Masters', 'PhD'.

- L'attributo 'Dept' indica il dipartimento in cui il lavoratore è impiegato al momento della recensione e può assumere i valori 'Administration', 'Engineering', 'Operations', 'Management', 'Sales'.
- L'attributo 'Seniority' rappresenta il livello di anzianità di servizio, ossia il numero di anni di esperienza, e può assumere valori interi compresi tra 1 e 5.

Infine, gli attributi 'Age', 'BasePay' e 'Bonus' sono quantitativi e rappresentano rispettivamente l'età del professionista, la sua paga annua in \$/anno e il suo bonus annuo in \$/anno.

4 - Diagramma delle classi principali e descrizione degli algoritmi coinvolti

4.1 Strutture dati

Il tool è stato realizzato seguendo le norme della buona progettazione che prevedono la separazione della logica applicativa dalla struttura dati e dall'interfaccia grafica. Pertanto il programma, scritto in linguaggio Java, segue i pattern MVC (Model View Controller) e DAO (Data Access Object).

Per la realizzazione dell'interfaccia grafica per l'utente è stato usato JavaFX e il software SceneBuilder.

I tre package realizzati sono:

- it.polito.tdp.Tesi, dove sono contenuti i Controller che ricevono i comandi dall'utente, li mandano alla logica applicativa e restituiscono i risultati, il Main che lancia l'esecuzione del programma e l'EntryPoint dove viene settata la scena.

```

  v it.polito.tdp.Tesi
    > AnalisiDatiController.java
    > EntryPoint.java
    > Main.java
    > RicercaPersonaleController.java

```

- it.polito.tdp.Tesi.db, dove viene gestita l'interazione con il database, ossia la connessione e le relative interrogazioni necessarie.

```

  v it.polito.tdp.Tesi.db
    > DBConnect.java
    > GenderGapDAO.java
    > TestDAO.java

```

- it.polito.tdp.Tesi.model, dove vi è tutta la logica applicativa, ossia tutti gli oggetti necessari e la classe 'Model' in cui vengono realizzati i metodi relativi all'elaborazione dei dati, come la ricorsione.

- ▼ it.polito.tdp.Tesi.model
 - > Model.java
 - > Professione.java
 - > TestModel.java
 - > Utente.java

4.2 Il package 'it.polito.tdp.Tesi'

All'interno del package 'it.polito.tdp.Tesi' sono presenti le due classi 'Main' e 'EntryPoint' che gestiscono il lancio del programma e le due classi che si occupano dell'interfaccia con l'utente. In particolare nella classe 'AnalisiDatiController' vi è il metodo che permette di ottenere il settore lavorativo selezionato dall'utente e ottenere i grafici per la visualizzazione delle metriche di interesse. Tale metodo si chiama 'Analizza' e si verifica quando l'utente effettua un click sull'omonimo bottone dell'interfaccia grafica.

A scopo illustrativo si riporta la creazione del grafico a torta riguardante la rappresentanza di genere e il grafico a barre per la differenza di stipendio medio annuo.

```
@FXML
void Analizza(ActionEvent event) {

    boolean updateReceived=true;
    txtResult.clear();
    txtInfo.clear();
    String jobTitle = this.cmbJobTitle.getValue();

    if(jobTitle==" " || jobTitle==null) {
        txtInfo.setStyle("-fx-text-fill: red; ");
        txtInfo.appendText("Seleziona un settore lavorativo!\n");
    }
    else {

        // creazione pie chart rappresentanza di genere

        this.model.Analizza(jobTitle);
        ObservableList<PieChart.Data> list1 = FXCollections.observableArrayList(
            new PieChart.Data("Percentuale uomini = "+this.model.getPercentualeMaschile()+" %", this.model.getPercentualeMaschile()),
            new PieChart.Data("Percentuale donne = "+this.model.getPercentualeFemminile()+" %", this.model.getPercentualeFemminile()));
        pieChart1.setData(list1);
        pieChart1.setClockwise(true);
        pieChart1.setLabelsVisible(true);
        pieChart1.setStartAngle(180);
        pieChart1.setLegendVisible(true);
        pieChart1.setVisible(true);
        pieChart1.setLabelLineLength(10);
        pieChart1.setLegendSide(Side.RIGHT);

        // creazione bar chart retribuzione annua media

        ObservableList<Series<String, Double>> allSeries1 = barChart1.getData();

        if (updateReceived) {
            for (XYChart.Series<String, Double> series : allSeries1) {
                for (XYChart.Data<String, Double> data : series.getData()) {
                    Node node = data.getNode();
                    Parent parent = node.parentProperty().get();
                    if (parent != null && parent instanceof Group) {
                        Group group = (Group) parent;
                        group.getChildren().clear();
                    }
                }
            }
            allSeries1.clear();
        }

        final String pagaAnnualeMedia = "Paga annua media";

        XYChart.Series<String, Double> series1 = new XYChart.Series<>();
        series1.setName("Uomini");
        final XYChart.Data<String, Double> data1 = new XYChart.Data<>(pagaAnnualeMedia, this.model.getAvgMalePay());
        data1.nodeProperty().addListener(new ChangeListener<Node>() {
            @Override public void changed(ObservableValue<? extends Node> ov, Node oldNode, final Node node) {
                if (node != null) {
                    displayLabelForData(data1);
                }
            }
        });
        series1.getData().add(data1);
    }
}
```

```

XYChart.Series<String, Double> series2 = new XYChart.Series<>();
series2.setName("Donne");
final XYChart.Data<String,Double> data2 = new XYChart.Data<>(pagaAnnualeMedia, this.model.getAvgFemalePay());
data2.nodeProperty().addListener(new ChangeListener<Node>() {
    @Override public void changed(ObservableValue<? extends Node> ov, Node oldNode, final Node node) {
        if (node != null) {
            displayLabelForData(data2);
        }
    }
});
series2.getData().add(data2);

barChart1.getData().addAll(Arrays.asList(series1, series2));
barChart1.setBarGap(2);
barChart1.setCategoryGap(10);
barChart1.setLegendVisible(true);
barChart1.setLegendSide(Side.RIGHT);
barChart1.setVisible(true);

```

Metodo Analizza() - AnalisiDatiController

Nella classe ‘RicercaPersonaleController’ invece vengono raccolti i dati selezionati dall’utente in merito alle professionalità necessarie alla creazione del team work e successivamente vengono restituiti i risultati degli algoritmi ricorsivi da visualizzare sull’interfaccia.

In seguito al selezionamento delle specifiche, tramite slider e comboBox, e al click dell’utente sul bottone denominato ‘Cerca Teamwork’ i dati selezionati vengono preparati e inviati alla classe Model in cui verranno elaborati e restituiti i risultati.

In seguito si riporta una parte del metodo che si occupa di verificare quali specifiche l’utente abbia richiesto.

```

@FXML
void cercaTeamWork(ActionEvent event) {

    txtResult.clear();
    txtField1.clear();
    txtField2.clear();
    tableView1.getItems().clear();
    tableView2.getItems().clear();
    pieChartRicorsione1.getData().clear();
    pieChartRicorsione2.getData().clear();
    int numDataScientist=0;
    int numDriver=0;
    int numFinancialAnalyst=0;
    int numGraphicDesigner=0;
    int numIT=0;
    int numManager=0;
    int numMarketingAssociate=0;
    int numSalesAssociate=0;
    int numSoftwareEngineer=0;
    int numWarehouseAssociate=0;

    ArrayList<Professione> professioniRicerca = new ArrayList<Professione>();

    if(this.sldDataScientist.getValue()!=0) {
        numDataScientist=(int) (this.sldDataScientist.getValue());
        if(this.cmbEdDS.getValue()==null || this.cmbSenDS.getValue()==null) {
            txtResult.appendText("Seleziona 'Education' e 'Seniority' per la posizione 'Data Scientist!'");
            return;
        }
        String education=this.cmbEdDS.getValue();
        int seniority = this.cmbSenDS.getValue();
        Professione p = new Professione("Data Scientist", seniority, education);

        for(int i=0; i<numDataScientist; i++) {
            professioniRicerca.add(p);
        }
    }
    else {
        this.cmbEdDS.setValue(null);
        this.cmbSenDS.setValue(null);
    }
}

```

Metodo cercaTeamWork() - RicercaPersonaleController

Nel seguente codice invece si ottengono i risultati dell'elaborazione dei dati da parte del primo algoritmo ricorsivo e si rappresentano in una TableView e in un grafico a torta.

```
List<Utente> utenti = model.cercaTeamWork(professioniRicercate);
this.tableView1.setItems(FXCollections.observableArrayList(utenti));
if(utenti.size()==0) {
    txtResult.setText("Non è stato possibile calcolare un team work con le specifiche richieste");
    txtResult.setStyle("-fx-text-fill: red;");
    return;
}
double percDonnel=this.calcolaPercentuali(utenti);
double percUomini=100-percDonnel;
ObservableList<PieChart.Data> list1 = FXCollections.observableArrayList(
    new PieChart.Data("Percentuale uomini = "+percUomini+" %", percUomini),
    new PieChart.Data("Percentuale donne = "+percDonnel+" %", percDonnel));
pieChartRicorsione1.setData(list1);
pieChartRicorsione1.setClockwise(true);
pieChartRicorsione1.setLabelsVisible(true);
pieChartRicorsione1.setStartAngle(180);
pieChartRicorsione1.setLegendVisible(true);
pieChartRicorsione1.setVisible(true);
pieChartRicorsione1.setLabelLineLength(10);
pieChartRicorsione1.setLegendSide(Side.RIGHT);
```

Metodo cercaTeamWork() - RicercaPersonaleController

4.3 Il package 'it.polito.tdp.Tesi.db'

Nel package 'it.polito.tdp.Tesi.db' avviene la connessione al database e la formulazione di tutte le interrogazioni necessarie all'estrazione delle informazioni. In particolare la connessione avviene nella classe 'DBConnect' attraverso il seguente metodo in cui è possibile specificare le proprie credenziali di accesso (campi 'username' e 'password').

```
public class DBConnect {

    private static String jdbcURL = "jdbc:mysql://localhost/gender_pay_gap";
    private static HikariDataSource ds;

    public static Connection getConnection() {

        if (ds == null) {
            HikariConfig config = new HikariConfig();
            config.setJdbcUrl(jdbcURL);
            config.setUsername("");
            config.setPassword("");

            // configurazione MySQL
            config.addDataSourceProperty("cachePrepStmts", "true");
            config.addDataSourceProperty("prepStmtCacheSize", "250");
            config.addDataSourceProperty("prepStmtCacheSqlLimit", "2048");

            ds = new HikariDataSource(config);
        }

        try {
            Connection c = ds.getConnection();
            return c;
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            return null;
        }
    }
}
```

Metodo getConnection() - DBConnect

Le interrogazioni al data-set vengono effettuate nella classe ‘GenderGapDAO’ tramite appositi metodi come quello riportato di seguito che permette di ottenere tutti i record presenti nella base di dati.

```
public List<Utente> getAll(){
    String sql = "SELECT * FROM gender_gap ";
    List<Utente> result = new ArrayList<Utente>();
    Connection conn = DBConnect.getConnection();

    try {
        PreparedStatement st = conn.prepareStatement(sql);
        ResultSet res = st.executeQuery();
        while (res.next()) {

            Utente gpg = new Utente(res.getInt("Id"),res.getString("JobTitle"),
                                    res.getString("Gender"),res.getInt("Age"),res.getInt("PerfEval"),
                                    res.getString("Education"),res.getString("Dept"),
                                    res.getInt("Seniority"),res.getInt("BasePay"),res.getInt("Bonus"));

            result.add(gpg);
        }
        res.close();
        st.close();
        conn.close();
        return result;
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}
```

Metodo getAll() - GenderGapDAO

Il seguente metodo invece permette di ottenere il numero di professionisti di genere maschile che lavorano nel settore selezionato dall’utente.

```
public int getMaleNumber(String jobTitle) {
    String sql = "SELECT COUNT(*) as tot FROM gender_gap WHERE jobTitle=? and Gender='Male' ";
    Connection conn = DBConnect.getConnection();
    int result;

    try {
        PreparedStatement st = conn.prepareStatement(sql);
        st.setString(1, jobTitle);
        ResultSet res = st.executeQuery();
        res.next();
        result = res.getInt("tot");
        res.close();
        st.close();
        conn.close();
        return result;
    } catch (SQLException e) {
        e.printStackTrace();
        return -1;
    }
}
```

Metodo getMaleNumber() – GenderGapDAO

Infine, nel package ‘it.polito.tdp.Tesi.db’ è presente una classe di supporto denominata ‘TestDAO’ utile unicamente per verificare, in fase di creazione del programma, il corretto funzionamento del codice qualora si sia ancora sprovvisti dell’interfaccia.

4.4 Il package 'it.polito.tdp.Tesi.model'

Nel package 'it.polito.tdp.Tesi.model' vi sono i vari oggetti, in particolare 'Utente' e 'Professione', e la classe Model dove avviene la vera e propria elaborazione dei dati.

L'oggetto 'Utente' riportato di seguito rappresenta i record presenti nella base di dati.

Conseguentemente è identificato da un numero intero univoco e pertanto due oggetti di tipo 'Utente' sono uguali se e solo se hanno lo stesso id.

```
public class Utente{  
  
    int id;  
    String jobTitle;  
    String gender;  
    int age;  
    int perfEval;  
    String education;  
    String dept;  
    int seniority;  
    int basePay;  
    int bonus;  
  
    public Utente(int id, String jobTitle, String gender, int age, int perfEval,  
                  String education, String dept, int seniority, int basePay, int bonus) {  
        this.id = id;  
        this.jobTitle = jobTitle;  
        this.gender = gender;  
        this.age = age;  
        this.perfEval = perfEval;  
        this.education = education;  
        this.dept = dept;  
        this.seniority = seniority;  
        this.basePay = basePay;  
        this.bonus = bonus;  
    }  
}
```

Classe Utente

L'oggetto 'Professione' invece è identificato da un sottoinsieme di attributi dell'oggetto 'Utente' ossia 'jobTitle', 'seniority', 'education'. Ciò significa che una professione è univocamente identificata da un titolo professionale, da un livello di anzianità di servizio e da un grado di istruzione. Si è realizzato l'oggetto in tale maniera per poter permettere all'utente di scegliere, per qualsiasi titolo professionale, gli altri due attributi in modo da soddisfare le esigenze del team work che si vuole realizzare.

```

public class Professione {

    String jobTtitle;
    int seniority;
    String education;

    public Professione(String jobTtitle, int seniority, String education) {
        super();
        this.jobTtitle = jobTtitle;
        this.seniority = seniority;
        this.education = education;
    }
}

```

Classe Professione

Nella classe ‘Model’ compare il metodo ‘Analizza’ che fornisce i dati da visualizzare tramite grafico nell’interfaccia alla classe ‘AnalisiDatiController’.

```

public void Analizza(String jobTitle) {

    this.totalNumber = this.gpgDAO.getTotalNumber(jobTitle);
    this.femaleNumber = this.gpgDAO.getFemaleNumber(jobTitle);
    this.maleNumber = this.gpgDAO.getMaleNumber(jobTitle);
    this.percentualeFemminile = ((double) (femaleNumber) / (double) (totalNumber)) * 100;
    this.percentualeMaschile = ((double) (maleNumber) / (double) (totalNumber)) * 100;
    this.avgFemalePay = this.gpgDAO.getAvgFemalePay(jobTitle).get(0);
    this.avgMalePay = this.gpgDAO.getAvgMalePay(jobTitle).get(0);
    this.avgFemaleBonus = this.gpgDAO.getAvgFemalePay(jobTitle).get(1);
    this.avgMaleBonus = this.gpgDAO.getAvgMalePay(jobTitle).get(1);
    this.avgFemaleAge = this.gpgDAO.getAvgFemaleAge(jobTitle);
    this.avgMaleAge = this.gpgDAO.getAvgMaleAge(jobTitle);
    this.avgFemaleSeniority = this.gpgDAO.getAvgFemaleSeniority(jobTitle);
    this.avgMaleSeniority = this.gpgDAO.getAvgMaleSeniority(jobTitle);
    this.percFemaleCollege = (double) (this.gpgDAO.getNumberFemaleEducation(jobTitle, "College")) / (double) (this.femaleNumber) * 100;
    this.percMaleCollege = (double) (this.gpgDAO.getNumberMaleEducation(jobTitle, "College")) / (double) (this.maleNumber) * 100;
    this.percFemaleHighSchool = (double) (this.gpgDAO.getNumberFemaleEducation(jobTitle, "High School")) / (double) (this.femaleNumber) * 100;
    this.percMaleHighSchool = (double) (this.gpgDAO.getNumberMaleEducation(jobTitle, "High School")) / (double) (this.maleNumber) * 100;
    this.percFemaleMasters = (double) (this.gpgDAO.getNumberFemaleEducation(jobTitle, "Masters")) / (double) (this.femaleNumber) * 100;
    this.percMaleMasters = (double) (this.gpgDAO.getNumberMaleEducation(jobTitle, "Masters")) / (double) (this.maleNumber) * 100;
    this.percFemalePhD = (double) (this.gpgDAO.getNumberFemaleEducation(jobTitle, "PhD")) / (double) (this.femaleNumber) * 100;
    this.percMalePhD = (double) (this.gpgDAO.getNumberMaleEducation(jobTitle, "PhD")) / (double) (this.maleNumber) * 100;
    this.percFemale1 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Female", 1)) / (double) (this.femaleNumber) * 100;
    this.percFemale2 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Female", 2)) / (double) (this.femaleNumber) * 100;
    this.percFemale3 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Female", 3)) / (double) (this.femaleNumber) * 100;
    this.percFemale4 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Female", 4)) / (double) (this.femaleNumber) * 100;
    this.percFemale5 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Female", 5)) / (double) (this.femaleNumber) * 100;

    this.percMale1 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Male", 1)) / (double) (this.maleNumber) * 100;
    this.percMale2 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Male", 2)) / (double) (this.maleNumber) * 100;
    this.percMale3 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Male", 3)) / (double) (this.maleNumber) * 100;
    this.percMale4 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Male", 4)) / (double) (this.maleNumber) * 100;
    this.percMale5 = (double) (this.gpgDAO.getNumPerfEvScore(jobTitle, "Male", 5)) / (double) (this.maleNumber) * 100;

    this.percFemaleSen1 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Female", 1)) / (double) (this.femaleNumber) * 100;
    this.percFemaleSen2 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Female", 2)) / (double) (this.femaleNumber) * 100;
    this.percFemaleSen3 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Female", 3)) / (double) (this.femaleNumber) * 100;
    this.percFemaleSen4 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Female", 4)) / (double) (this.femaleNumber) * 100;
    this.percFemaleSen5 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Female", 5)) / (double) (this.femaleNumber) * 100;
    this.percMaleSen1 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Male", 1)) / (double) (this.maleNumber) * 100;
    this.percMaleSen2 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Male", 2)) / (double) (this.maleNumber) * 100;
    this.percMaleSen3 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Male", 3)) / (double) (this.maleNumber) * 100;
    this.percMaleSen4 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Male", 4)) / (double) (this.maleNumber) * 100;
    this.percMaleSen5 = (double) (this.gpgDAO.getNumSeniority(jobTitle, "Male", 5)) / (double) (this.maleNumber) * 100;
}

```

Metodo Analizza() - Model

Inoltre, sono presenti anche i due algoritmi ricorsivi necessari alla generazione del team-work con il massimo ‘Performance Evaluation Score’. Nel metodo ‘cercaTeamWork’ vengono inizializzate le variabili necessarie ossia:

- vengono caricati tutti gli utenti nell’ArrayList denominata ‘utenti’
- le ArrayList chiamata ‘best’ e ‘parziale’ vengono inizializzate a liste vuote di oggetti di tipo ‘Utente’

- le ArrayList 'professioniRicerca' e 'professioniRicercaModificabile' vengono caricate con gli oggetti di tipo 'Professione' selezionati dall'utente
- viene inizializzato un iteratore 'itr' che permette di scorrere la lista degli utenti e eliminare quelli che non rispettano le specifiche, ossia non hanno un titolo professionale richiesto o non hanno il grado di anzianità o istruzione specificato.

```
public List<Utente> cercaTeamWork(List<Professione> professioniRicerca) {
    this.utenti = this.gpgDAO.getAll();
    this.best=new ArrayList<Utente>();
    List<Utente> parziale = new ArrayList<Utente>();
    this.professioniRicerca = professioniRicerca;
    List<Professione> professioniRicercaModificabile = new ArrayList<Professione>(professioniRicerca);
    Iterator<Utente> itr = utenti.iterator();

    while(itr.hasNext()) {
        Utente u = itr.next();
        if(!professioniRicerca.contains(new Professione(u.getJobTitle(),u.getSeniority(),u.getEducation()))) {
            itr.remove();
        }
    }

    System.out.println("Numero utenti : "+this.utenti.size()+"\n");

    ricorsione(parziale, professioniRicercaModificabile,0);

    System.out.println("Ricorsione terminata!\n");
    for(Utente u : this.best) {
        System.out.println(u+"\n");
    }

    return best;
}
```

Metodo cercaTeamWork() - Model

Nel metodo 'ricorsione' avviene il vero e proprio algoritmo ricorsivo che viene realizzato in diversi passaggi :

- 1) Inizialmente si controlla se la lista 'professioniRicercaModificabile' abbia ancora qualche elemento al suo interno. Se la lista non ha più elementi al suo interno allora si calcola la somma dei 'Performance Evaluation Score' degli utenti presenti in 'parziale' e si confronta il risultato con il migliore risultato ottenuto che è presente nella lista 'best'. Se il risultato è migliore allora si aggiorna la lista 'best' e si salva al suo interno 'parziale'. Altrimenti si ritorna alla funzione chiamante e si procede all'eliminazione di un elemento.
- 2) Successivamente si va a controllare di non aver già effettuato la scansione di tutti gli utenti presenti nella lista 'utenti'. Se tutti gli utenti fossero già stati visualizzati allora la 'return' richiama la funzione chiamante.
- 3) Il terzo controllo permette di terminare in tempi più rapidi la ricorsione qualora gli utenti selezionati nella lista 'best' abbiano tutti un 'performance Evaluation Score' massimo, ossia pari a 5.
- 4) Infine, se ci sono ancora dei professionisti da aggiungere al team work si procede a verificare se il successivo elemento presente nella lista 'utenti' possa essere aggiunto e, se la risposta è positiva, si prova ad aggiungere andando al successivo livello di ricorsione e poi si prova a non aggiungere procedendo comunque con l'algoritmo. Se la risposta è negativa si procede ugualmente con la ricerca del miglior team.

```

private void ricorsione(List<Utente> parziale, List<Professione> professioniRicercaModificabile, int livello){

    // condizione di terminazione
    if(professioniRicercaModificabile.size()==0) {
        // controllo se tale soluzione è la migliore
        if(calcolaTotScore(parziale)>calcolaTotScore(best)) {
            this.best = new ArrayList<Utente>(parziale);
            System.out.println(calcolaTotScore(best)+"\n");
            for(Utente u : this.best) {
                System.out.println(u+"\n");
            }
        }
        return;
    }

    if(this.utenti.size()==livello) {
        return;
    }

    if(calcolaTotScore(best)==(this.professioniRicerca.size()*5)) {
        return;
    }

    if(professioniRicercaModificabile.contains(new Professione(utenti.get(livello).getJobTitle(),
                                                                utenti.get(livello).getSeniority(),
                                                                utenti.get(livello).getEducation())) {

        //provo ad aggiungere
        parziale.add(utenti.get(livello));
        professioniRicercaModificabile.remove(new Professione(utenti.get(livello).getJobTitle(),
                                                                utenti.get(livello).getSeniority(),
                                                                utenti.get(livello).getEducation()));

        ricorsione(parziale,professioniRicercaModificabile, livello+1);

        // provo a non aggiungere
        professioniRicercaModificabile.add(new Professione(parziale.get(parziale.size()-1).getJobTitle(),
                                                            parziale.get(parziale.size()-1).getSeniority(),
                                                            parziale.get(parziale.size()-1).getEducation()));

        parziale.remove(parziale.size()-1);
        ricorsione(parziale,professioniRicercaModificabile, livello+1);
    }

    ricorsione(parziale,professioniRicercaModificabile, livello+1);
}

```

Metodo ricorsione() – Model

Il secondo algoritmo ricorsivo funziona in maniera molto simile con la differenza che la condizione di terminazione prevede che l'algoritmo debba trovare il team work che massimizzi il 'Performance Evaluation Score' e che mantenga più equa possibile la rappresentanza di genere.

```

private void ricorsioneEqua(List<Utente> parziale, List<Professione> professioniRicercaModificabile, int livello){

    // condizione di terminazione
    if(professioniRicercaModificabile.size()==0) {
        int score = calcolaTotScore(parziale);
        // controllo se tale soluzione è la migliore
        if(score>calcolaTotScore(best) && soluzioneEqua(parziale)) {
            this.best = new ArrayList<Utente>(parziale);
            System.out.println(calcolaTotScore(best)+"\n");
            for(Utente u : this.best) {
                System.out.println(u+"\n");
            }
        }
        return;
    }

    if(this.utenti.size()==livello) {
        return;
    }

    if(calcolaTotScore(best)==(this.professioniRicerca.size()*5)) {
        return;
    }

    if(professioniRicercaModificabile.contains(new Professione(utenti.get(livello).getJobTitle(),
                                                                utenti.get(livello).getSeniority(),
                                                                utenti.get(livello).getEducation())) {

```



```

//provo ad aggiungere
parziale.add(utenti.get(livello));
professioniRicercaModificabile.remove(new Professione(utenti.get(livello).getJobTitle(),
                                                         utenti.get(livello).getSeniority(),
                                                         utenti.get(livello).getEducation()));

ricorsioneEqua(parziale,professioniRicercaModificabile, livello+1);

// provo a non aggiungere
professioniRicercaModificabile.add(new Professione(parziale.get(parziale.size()-1).getJobTitle(),
                                                    parziale.get(parziale.size()-1).getSeniority(),
                                                    parziale.get(parziale.size()-1).getEducation()));

parziale.remove(parziale.size()-1);
ricorsioneEqua(parziale,professioniRicercaModificabile, livello+1);
}

ricorsioneEqua(parziale,professioniRicercaModificabile, livello+1);
}

```

Metodo ricorsioneEqua() - Model

Il metodo che permette all'algoritmo di trovare un team con un'equa rappresentanza di genere richiede che il numero di donne presenti nella soluzione sia pari alla divisione intera per 2 del numero totale di professionisti richiesti nel team. In tal modo se il numero di persone nel team fosse dispari si richiederebbe comunque una presenza femminile significativa.

```

private boolean soluzioneEqua(List<Utente> parziale) {
    int numDonne=0;

    for(Utente u : parziale) {
        if(u.getGender().equals("Female")) {
            numDonne++;
        }
    }

    if(numDonne==parziale.size()/2) {
        return true;
    }
    else {
        return false;
    }
}

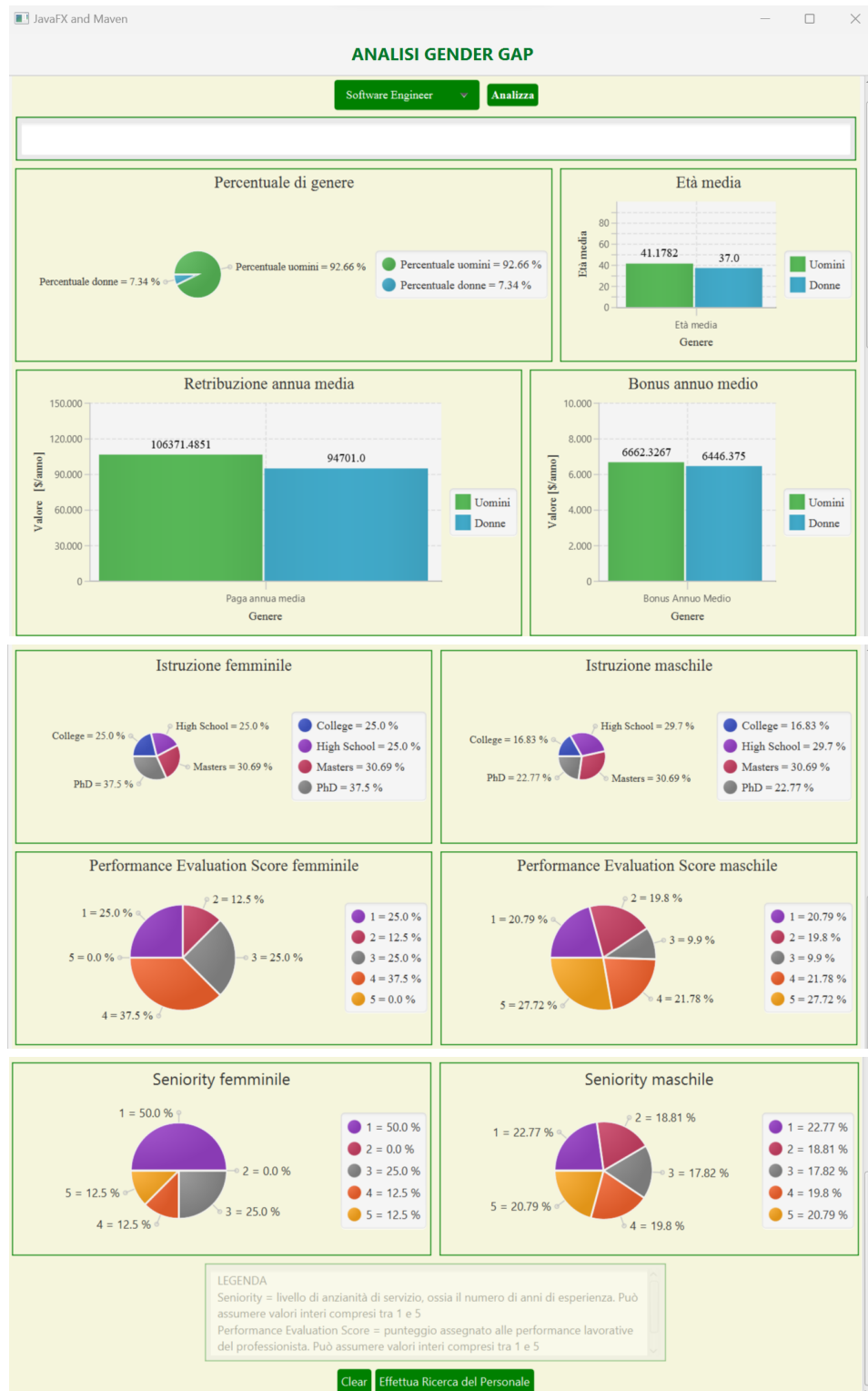
```

Metodo soluzioneEqua() - Model

5 – Esempio di utilizzo dell'applicazione

5.1 Esempio analisi dei dati

Ipotizzando che l'utente voglia vedere effettivamente quale sia, all'interno di questo data-set, la distribuzione di genere per il titolo professionale di 'Software Engineer', questo è ciò che otterrebbe.



5.2 Esempio ricerca del personale

Successivamente, se effettuasse la ricerca di un team con :

- 3 professionisti nel settore IT con grado di anzianità di servizio pari a 4 e titolo di studio pari ad un Master
- 2 manager con grado di anzianità di servizio pari a 2 e titolo di studio pari alla High School

- 3 Software Engineer con grado di anzianità di servizio pari a 3 e titolo di studio pari ad un PhD

La soluzione sarebbe la seguente:

JavaFX and Maven


RICERCA PERSONALE

SETTORE	QUANTITÀ	SENIORITY	EDUCATION
DATA SCIENTIST	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
DRIVER	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
FINANCIAL ANALYST	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
GRAPHIC DESIGNER	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
IT	<input type="range" value="3"/>	4 <input type="button" value="▼"/>	Masters <input type="button" value="▼"/>
MANAGER	<input type="range" value="2"/>	2 <input type="button" value="▼"/>	High School <input type="button" value="▼"/>
MARKETING ASSOCIATE	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
SALES ASSOCIATE	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>
SOFTWARE ENGINEER	<input type="range" value="3"/>	3 <input type="button" value="▼"/>	PhD <input type="button" value="▼"/>
WAREHOUSE ASSOCIATE	<input type="range" value="0"/>	<input type="button" value="▼"/>	<input type="button" value="▼"/>

SOLUZIONE LIBERA

ID	JOB TITLE	GENDER	AGE	PERFORM...	EDUCA
64	Software E...	Male	19	5	PhD
160	IT	Male	18	4	Maste

Rappresentanza di genere



● Percentuale uomini = 75.0 %


● Percentuale donne = 25.0 %

Performance Evaluation Score medio : 3.88

SOLUZIONE EQUIGENERE

ID	JOB TITLE	GENDER	AGE	PERFORM...	EDUCA
64	Software E...	Male	19	5	PhD
160	IT	Male	18	4	Maste

Rappresentanza di genere



● Percentuale uomini = 50.0 %

● Percentuale donne = 50.0 %

Performance Evaluation Score medio : 3.25

5.3 Link al video YouTube

È disponibile un video illustrativo del funzionamento dell'applicazione al seguente link :

<https://www.youtube.com/watch?v=OoIEJGWtNPE>

16 – Valutazione dei risultati e conclusioni

16.1 Valutazione dei risultati

L'obiettivo primario di permettere una lettura più immediata ed efficace di alcuni dati sulle differenze di genere è stato implementato. Poiché il data-set ha un numero limitato di record, i risultati ottenuti non rappresentano la reale situazione in termini di disparità di genere, tuttavia è possibile notare alcuni trend che rappresentano abbastanza fedelmente l'attualità. Inoltre, qualora si disponesse di maggiori dati, sarebbe possibile ugualmente visualizzarli ed elaborarli nell'applicazione.

Per quanto riguarda la seconda parte dell'applicazione le criticità riguardano sicuramente il fatto che, selezionando molti titoli di lavoro, i tempi computazionali per eseguire l'algoritmo ricorsivo aumentano notevolmente per via dell'aumento del numero di record. Tuttavia, la problematica può essere ovviata semplicemente scomponendo il team ricercato in più team ed effettuando più ricerche. Ciò che emerge è che, cercando di generare un team equi-genere, può capitare che ciò non sia possibile per la mancanza di professionisti di un certo genere in un determinato settore. Ad esempio, nel settore degli ingegneri del software, su 109 professionisti presenti solo 9 sono di genere femminile, di cui solo una con punteggio relativo alle performance massimo. Tale situazione compromette significativamente la possibilità di generare un team con la più equa rappresentanza di genere. Un'altra evidenza che emerge è che, nella maggior parte dei casi, il team con maggiore equità di genere presenta una somma dei punteggi relativi alle performance dei lavoratori leggermente inferiore rispetto al team generato con l'unico obiettivo di massimizzare tale punteggio. È nelle due situazioni appena descritte che si inseriscono i concetti di segregazione orizzontale e verticale intese come sotto-rappresentanza di un determinato genere in alcuni settori lavorativi e mancanza di persone di un determinato genere nelle posizioni dirigenziali di maggiore prestigio.

16.2 Conclusioni

L'obiettivo di permettere all'utente di familiarizzare con dati di natura ridondante e difficilmente leggibile è stato raggiunto. L'utilizzo dei grafici rende la lettura dell'applicazione semplificata e interattiva. Inoltre, la funzione di ricerca del personale risulta essere, oltre che utile, particolarmente impattante in quanto permette di vedere graficamente e in maniera immediata le differenze tra un team e l'altro in termini di rappresentanza di genere. In futuro potrebbero essere implementate nuove funzionalità che permettano di effettuare analisi più accurate, come inferenze statistiche, che possano consentire di individuare quanto questi dati realmente rispecchino la realtà. Diversamente, potrebbero essere utilizzati data-set più numerosi e accurati che rappresentino meglio la situazione reale.

Questa relazione tecnica è rilasciata con licenza Creative Commons BY-NC-SA.

Tu sei libero di:

- Condividere - riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare questo materiale con qualsiasi mezzo e formato
- Modificare - remixare, trasformare il materiale e basarti su di esso per le tue opere

Alle seguenti condizioni:

- Attribuzione - Devi riconoscere una menzione di paternità adeguata, fornire un link alla licenza e indicare se sono state effettuate delle modifiche. Puoi fare ciò in qualsiasi maniera ragionevole possibile, ma non con modalità tali da suggerire che il licenziante avalli te o il tuo utilizzo del materiale.
- Non Commerciale - Non puoi utilizzare il materiale per scopi commerciali.
- StessaLicenza - Se remixi, trasformi il materiale o ti basi su di esso, devi distribuire i tuoi contributi con la stessa licenza del materiale originario.

Per visualizzare una copia di questa licenza, visitare : <https://creativecommons.org/licenses/by-nc-sa/4.0/>

