Es. 11 (dal Dataset all'algoritmo)

April 10, 2024

1 DAL DATASET ALL'ALGORITMO: COME SI SVILLUPA UN MODELLO?

In questa esercitazione viene mostrato come da un semplice Dataset (in questo caso scaricato da Internet dalla community di Kaggle, https://www.kaggle.com/?utm_source=homescreen) si riesce a sviluppare un modello (quindi un algoritmo) per prevedere una o più variabili target. Per fare questo bisogna prima però eseguire dei passaggi preliminari che son fondamentali per la cura e la precisione del modello finale (come ad esempio quelli di gestire i NaN e gli Outliers)

Il mio dataset è questo: https://www.kaggle.com/datasets/arnabchaki/data-science-salaries-2023

1.1 FASE 1: SCEGLIERE (O CREARE), IMPORTARE E SALVARE IL DATASET

1) SCARICARE IL DATASET E INSERIRLO IN UN PATH (PER COMODITA LO METTO NELLA STESSA CARTELLA)
2) IMPORTARE LE LIBRERIE NECESSARIE: PANDAS (PER LEGGERE IL DATASET) E OS (PER GESTIRE I PATH)
3) IMPORTARE IL DATASET USANDO LE FUNZIONI DI PANDAS

[ ]:
```
import pandas  as pd # Importare la libreria "Pandas" per poter gestire i␣
 ↪Dataset
import os # Importare la libreria "os" per gestire i path

# Per importare il Dataset possiamo usare due funzione di Pandas:
# 1) pd.read_csv(): per leggere il file CSV (comma separated values)
# 2) pd.read_excel(): per leggere i file Excel

path_dataset = r"C:\Users\matte\OneDrive - Scuola Paritaria S. Freud␣
 ↪SRL\Desktop\FREUD\2D\QUADERNI E ALTRO\ROBOTICA ED AI\ESERCIZI IN CLASSE␣
 ↪PYTHON\ds_salaries.csv" # Il prefisso "r" serve per evitare che ci siano␣
 ↪confusioni nell'interpretazione della stringa, come ad esempio: numeri,␣
 ↪caratteri speciali e backslash
dataset = pd.read_csv(path_dataset)
```

1.2 FASE 2: VISUALIZZAZIONE E ANALISI DEL DATASET (CON I GRAFICI)

1) STAMPARE IL DATASET
2) PER OGNI FEATURE ANALIZZARE COME SIA COMPOSTA: CIOè CHE VALORI HA NEL DETTAGLIO (TIPO UNITà DI MISURA O VALUTE)
3) ANALIZZARE COSA SIA MEGLIO TENERE O COSA INVECE è MEGLIO BUTTARE

- **Esperienza Lavorativa:**
  - Questa feature indica il livello di esperienza lavorativa del candidato. Può assumere i seguenti valori:
    * SE (Senior)
    * MI (Mid-level)
    * EN (Entry-level)
- **Tipo di Impiego:**
  - Questa feature specifica il tipo di impiego svolto dal candidato. Può essere:
    * FT (Full-time)
    * CT (Contract)

[ ]:
```
dataset # Stampare il Dataset serve per poterlo analizzare nel dettaglio␣
↪meglio, come ad esempio visualizzare le Feature e le istanze per decidere␣
↪cose sia meglio tenere e cosa invece sia meglio eliminare
# Scrivendo solo il nome del dataset, quest'ultimo si stamperà (solo la parte␣
↪iniziale e finale)
```

[ ]:
```
      work_year experience_level employment_type                 job_title  \
0          2023               SE              FT  Principal Data Scientist
1          2023               MI              CT                ML Engineer
2          2023               MI              CT                ML Engineer
3          2023               SE              FT             Data Scientist
4          2023               SE              FT             Data Scientist
…           …                …               …                          …
3750       2020               SE              FT             Data Scientist
3751       2021               MI              FT  Principal Data Scientist
3752       2020               EN              FT             Data Scientist
3753       2020               EN              CT      Business Data Analyst
3754       2021               SE              FT       Data Science Manager

       salary salary_currency  salary_in_usd employee_residence  remote_ratio  \
0       80000             EUR          85847                ES           100
1       30000             USD          30000                US           100
2       25500             USD          25500                US           100
3      175000             USD         175000                CA           100
4      120000             USD         120000                CA           100
…       …                …               …                 …             …
3750   412000             USD         412000                US           100
3751   151000             USD         151000                US           100
3752   105000             USD         105000                US           100
```

```
3753   100000             USD         100000                  US           100
3754  7000000             INR          94665                  IN            50

     company_location company_size
0                  ES            L
1                  US            S
2                  US            S
3                  CA            M
4                  CA            M
…                 …             …
3750               US            L
3751               US            L
3752               US            S
3753               US            L
3754               IN            L

[3755 rows x 11 columns]
```

[ ]:
```
# Stampare i valori unici (unique), nonchè tutti i possibili output per ogni␣
 ↪Feature, serve per analizzare meglio il Dataset nel dettaglio di ogni␣
 ↪Feature e capire così tutti i possibili ambiti
print("I valori di work_year sono:") # All'inizio viene stampata una stringa di␣
 ↪testo esplicativa
print(dataset["work_year"].unique()) # Poi si stampano i veri e propri valori␣
 ↪unici
print("I valori di experience_level sono:")
print(dataset["experience_level"].unique())
print("I valori di employment_type sono:")
print(dataset["employment_type"].unique())
print("I valori di job_title sono:")
print(dataset["job_title"].unique())
print("I valori di salary sono:")
print(dataset["salary"].unique())
print("I valori di salary_currency sono:")
print(dataset["salary_currency"].unique())
print("I valori di salary_in_usd sono:")
print(dataset["salary_in_usd"].unique())
print("I valori di employee_residence sono:")
print(dataset["employee_residence"].unique())
print("I valori di remote_ratio sono:")
print(dataset["remote_ratio"].unique())
print("I valori di company_location sono:")
print(dataset["company_location"].unique())
print("I valori di company_size sono:")
print(dataset["company_size"].unique())
```

```
I valori di work_year sono:
```

```
[2023 2022 2020 2021]
I valori di experience_level sono:
['SE' 'MI' 'EN' 'EX']
I valori di employment_type sono:
['FT' 'CT' 'FL' 'PT']
I valori di job_title sono:
['Principal Data Scientist' 'ML Engineer' 'Data Scientist'
 'Applied Scientist' 'Data Analyst' 'Data Modeler' 'Research Engineer'
 'Analytics Engineer' 'Business Intelligence Engineer'
 'Machine Learning Engineer' 'Data Strategist' 'Data Engineer'
 'Computer Vision Engineer' 'Data Quality Analyst'
 'Compliance Data Analyst' 'Data Architect'
 'Applied Machine Learning Engineer' 'AI Developer' 'Research Scientist'
 'Data Analytics Manager' 'Business Data Analyst' 'Applied Data Scientist'
 'Staff Data Analyst' 'ETL Engineer' 'Data DevOps Engineer' 'Head of Data'
 'Data Science Manager' 'Data Manager' 'Machine Learning Researcher'
 'Big Data Engineer' 'Data Specialist' 'Lead Data Analyst'
 'BI Data Engineer' 'Director of Data Science'
 'Machine Learning Scientist' 'MLOps Engineer' 'AI Scientist'
 'Autonomous Vehicle Technician' 'Applied Machine Learning Scientist'
 'Lead Data Scientist' 'Cloud Database Engineer' 'Financial Data Analyst'
 'Data Infrastructure Engineer' 'Software Data Engineer' 'AI Programmer'
 'Data Operations Engineer' 'BI Developer' 'Data Science Lead'
 'Deep Learning Researcher' 'BI Analyst' 'Data Science Consultant'
 'Data Analytics Specialist' 'Machine Learning Infrastructure Engineer'
 'BI Data Analyst' 'Head of Data Science' 'Insight Analyst'
 'Deep Learning Engineer' 'Machine Learning Software Engineer'
 'Big Data Architect' 'Product Data Analyst'
 'Computer Vision Software Engineer' 'Azure Data Engineer'
 'Marketing Data Engineer' 'Data Analytics Lead' 'Data Lead'
 'Data Science Engineer' 'Machine Learning Research Engineer'
 'NLP Engineer' 'Manager Data Management' 'Machine Learning Developer'
 '3D Computer Vision Researcher' 'Principal Machine Learning Engineer'
 'Data Analytics Engineer' 'Data Analytics Consultant'
 'Data Management Specialist' 'Data Science Tech Lead'
 'Data Scientist Lead' 'Cloud Data Engineer' 'Data Operations Analyst'
 'Marketing Data Analyst' 'Power BI Developer' 'Product Data Scientist'
 'Principal Data Architect' 'Machine Learning Manager'
 'Lead Machine Learning Engineer' 'ETL Developer' 'Cloud Data Architect'
 'Lead Data Engineer' 'Head of Machine Learning' 'Principal Data Analyst'
 'Principal Data Engineer' 'Staff Data Scientist' 'Finance Data Analyst']
I valori di salary sono:
[   80000    30000    25500   175000   120000   222200   136000   219000
   141000   147100    90700   130000   100000   213660   130760   170000
   150000   110000   275000   174000   230000   143200   225000   156400
   200000    90000    72000   253200   342810   184590   162500   105380
    64500  1650000   204620   110680   270703   221484   212750   185000
   262000   245000   275300   183500   218500   199098   203300   123600
```

189110 139000 258750 231500 166000 172500 110500 238000
176000 237000 201450 309400 159100 115000 81500 280000
210000 280100 168100 193500 510000 65000 300000 185900
129300 140000 45000 36000 105000 70000 163196 145885
217000 202800 104300 145000 165000 132300 179170 94300
152500 116450 247300 133800 203000 133000 220000 54000
289800 214000 179820 143860 283200 188800 214200 252000
129000 155000 161800 141600 342300 176100 85000 138784
83270 75000 204500 138900 318300 212200 95000 195000
160000 1700000 38000 35000 168400 105200 190000 241000
55000 15000 47500 250000 228000 186000 180000 50000
205000 215000 247500 172200 224000 1400000 128000 329500
269600 203500 152000 239000 122900 191765 134236 112000
84000 135000 105500 293000 148500 240500 123700 152900
117100 173000 113000 260000 184000 149500 127075 219535
146115 199000 162000 221000 153000 187000 179000 109000
142000 198800 125000 86000 106000 280700 150450 250500
159500 130001 71907 93918 51962 257000 147000 222000
133200 156000 304000 161200 84570 240000 183600 289076
202353 157750 104650 68000 60000 181000 154000 146000
64200 56100 208450 170550 171250 113750 153600 100500
182500 121500 203100 114500 92700 61800 258000 167500
106500 57000 286000 207000 223250 178600 353200 249300
297300 198200 151800 317070 170730 20000 108000 134000
124000 124500 148700 125600 120250 183000 1500000 216000
143865 115092 132000 208049 128500 149600 102000 106800
151000 7000 40000 143000 42000 111000 265000 235000
60400 164000 56000 83500 52500 201036 134024 62000
58000 172000 163800 126000 139500 109400 205600 105700
239748 159832 186300 102500 149040 113900 172600 107900
180180 106020 376080 213120 206500 121600 194500 115500
115934 81666 206000 138000 92000 48000 87000 299500
245100 115100 73900 141288 94192 210914 116704 185700
169000 110600 193000 136850 276000 178500 161000 83300
112700 128750 106250 188500 117000 104500 127000 94000
210550 153300 161500 119500 148750 146300 153400 122700
123900 340000 121700 310000 149076 82365 85500 97750
201000 122000 116990 82920 142200 205920 171600 78000
116000 36050 34320 93800 67000 1300000 1000000 104000
152380 121904 128280 106900 192000 170500 60027 44737
131899 104891 124740 65488 72200 64980 179975 86466
168000 167580 87980 202000 148000 269000 158000 197000
290000 172800 300240 200160 370000 137500 323300 184700
153088 183310 144000 66000 126277 126500 272000 259000
101400 288000 215050 198000 114000 209300 182200 227000
52000 226700 133300 124999 800000 63000 253750 169200
213580 163625 12000 375000 1350000 231250 138750 284310
153090 225900 385000 93919 241871 133832 192500 216100

140800 284000 236000 248100 145900 155850 102544 151410
115360 1050000 25000 107000 23000 182750 314100 195800
350000 262500 209450 158677 103200 61200 59000 174500
107250 119000 285800 154600 5000000 124234 74540 79000
141290 74178 107500 1060000 6000 1440000 840000 1250000
182000 234100 223800 172100 232200 167200 291500 196200
150900 167000 96100 196000 126100 187500 24000 165750
89700 55250 175308 100706 229000 4000000 272550 64000
143100 180560 115440 1125000 261500 134500 1100000 94500
127500 51000 248400 4460000 149000 246000 10000 2500000
2800000 249500 149850 122500 102640 66100 122600 159000
255000 166700 194000 129400 89200 178750 197430 134760
99000 105120 75360 171000 13000 213000 227200 61000
243000 178000 96000 137000 189750 140250 191200 179500
26000 118000 177000 131000 193750 116250 208000 45555
6600000 140700 33000 154560 123648 177500 192564 144854
179305 142127 315000 243900 156600 77300 45600 184100
198440 47000 187200 116100 159699 138938 76000 125404
123000 92250 97000 157000 345600 230400 175950 130050
236600 27000 400000 8000 123400 88100 139600 85700
98200 98000 144200 3000000 188700 160395 191475 141525
156868 178800 132100 229998 154545 99750 68400 236900
159200 243225 179775 218000 145300 195400 131300 195700
130500 141300 102100 83000 1800000 633000 179400 193900
222640 182160 297500 93000 73000 40300 136994 101570
97500 212800 142800 500000 130240 83376 65004 84958
66822 81000 46000 204100 136100 7500 77000 28500
119300 146200 124270 185800 137400 148800 7500000 82000
32400 216200 144100 175100 189650 164996 99450 188100
139860 248700 167100 450000 189500 140100 177600 202900
900000 4200000 260500 73400 49500 2400000 206699 99100
221300 74000 249260 185400 128875 93700 136260 109280
150075 110925 22800 112900 90320 62500 105400 43200
215300 158200 209100 165400 132320 208775 147800 6000000
100800 140400 82900 63900 112300 108800 242000 165220
120160 124190 181940 220110 160080 106260 120600 84900
136620 99360 161342 137141 211500 138600 192400 61300
95550 136600 167875 205300 200100 70500 116150 99050
192600 266400 150260 69000 324000 185100 104890 53000
88000 66500 121000 29000 69999 52800 405000 380000
8500000 7000000 38400 82500 700000 8760 51999 41000
13400 103000 270000 45760 44000 2250000 37456 11000000
14000 2200000 188000 2100000 51400 61500 720000 31000
91000 1600000 256000 72500 65720 111775 93150 21600
4900000 1200000 21000 1799997 9272 120500 21844 22000
76760 1672000 420000 30400000 32000 416000 40900 4450000
423000 325000 34000 69600 435000 37000 19000 18000
39600 1335000 1450000 190200 138350 130800 412000]

```
I valori di salary_currency sono:
['EUR' 'USD' 'INR' 'HKD' 'CHF' 'GBP' 'AUD' 'SGD' 'CAD' 'ILS' 'BRL' 'THB'
 'PLN' 'HUF' 'CZK' 'DKK' 'JPY' 'MXN' 'TRY' 'CLP']
I valori di salary_in_usd sono:
[ 85847  30000  25500 …  28369 412000  94665]
I valori di employee_residence sono:
['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'PT' 'NL' 'CH' 'CF' 'FR' 'AU'
 'FI' 'UA' 'IE' 'IL' 'GH' 'AT' 'CO' 'SG' 'SE' 'SI' 'MX' 'UZ' 'BR' 'TH'
 'HR' 'PL' 'KW' 'VN' 'CY' 'AR' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK'
 'IT' 'MA' 'LT' 'BE' 'AS' 'IR' 'HU' 'SK' 'CN' 'CZ' 'CR' 'TR' 'CL' 'PR'
 'DK' 'BO' 'PH' 'DO' 'EG' 'ID' 'AE' 'MY' 'JP' 'EE' 'HN' 'TN' 'RU' 'DZ'
 'IQ' 'BG' 'JE' 'RS' 'NZ' 'MD' 'LU' 'MT']
I valori di remote_ratio sono:
[100   0  50]
I valori di company_location sono:
['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'NL' 'CH' 'CF' 'FR' 'FI' 'UA'
 'IE' 'IL' 'GH' 'CO' 'SG' 'AU' 'SE' 'SI' 'MX' 'BR' 'PT' 'RU' 'TH' 'HR'
 'VN' 'EE' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK' 'IT' 'MA' 'PL' 'AL'
 'AR' 'LT' 'AS' 'CR' 'IR' 'BS' 'HU' 'AT' 'SK' 'CZ' 'TR' 'PR' 'DK' 'BO'
 'PH' 'BE' 'ID' 'EG' 'AE' 'LU' 'MY' 'HN' 'JP' 'DZ' 'IQ' 'CN' 'NZ' 'CL'
 'MD' 'MT']
I valori di company_size sono:
['L' 'S' 'M']
```

[ ]:
```python
print("Numero di occorrenze per ogni valore di work_year:")
print(dataset["work_year"].value_counts())
print("\nNumero di occorrenze per ogni valore di experience_level:")
print(dataset["experience_level"].value_counts())
print("\nNumero di occorrenze per ogni valore di employment_type:")
print(dataset["employment_type"].value_counts())
print("\nNumero di occorrenze per ogni valore di job_title:")
print(dataset["job_title"].value_counts())
print("\nNumero di occorrenze per ogni valore di salary:")
print(dataset["salary"].value_counts())
print("\nNumero di occorrenze per ogni valore di salary_currency:")
print(dataset["salary_currency"].value_counts())
print("\nNumero di occorrenze per ogni valore di salary_in_usd:")
print(dataset["salary_in_usd"].value_counts())
print("\nNumero di occorrenze per ogni valore di employee_residence:")
print(dataset["employee_residence"].value_counts())
print("\nNumero di occorrenze per ogni valore di remote_ratio:")
print(dataset["remote_ratio"].value_counts())
print("\nNumero di occorrenze per ogni valore di company_location:")
print(dataset["company_location"].value_counts())
print("\nNumero di occorrenze per ogni valore di company_size:")
print(dataset["company_size"].value_counts())
```

```
Numero di occorrenze per ogni valore di work_year:
```

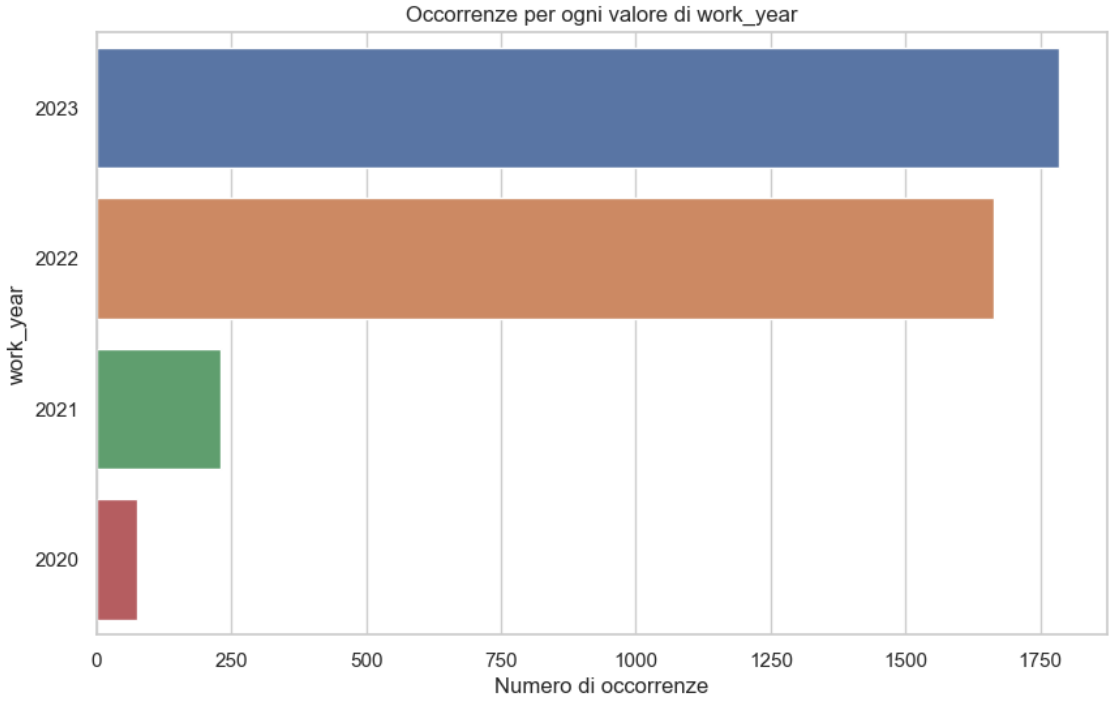
```
2023    1785
2022    1664
2021     230
2020      76
Name: work_year, dtype: int64

Numero di occorrenze per ogni valore di experience_level:
SE    2516
MI     805
EN     320
EX     114
Name: experience_level, dtype: int64

Numero di occorrenze per ogni valore di employment_type:
FT    3718
PT      17
CT      10
FL      10
Name: employment_type, dtype: int64

Numero di occorrenze per ogni valore di job_title:
Data Engineer                            1040
Data Scientist                            840
Data Analyst                              612
Machine Learning Engineer                 289
Analytics Engineer                        103
                                          …
Principal Machine Learning Engineer         1
Azure Data Engineer                         1
Manager Data Management                     1
Marketing Data Engineer                     1
Finance Data Analyst                        1
Name: job_title, Length: 93, dtype: int64

Numero di occorrenze per ogni valore di salary:
100000    112
150000    100
120000     99
160000     85
130000     85
          …
241871      1
93919       1
385000      1
225900      1
412000      1
Name: salary, Length: 815, dtype: int64
```

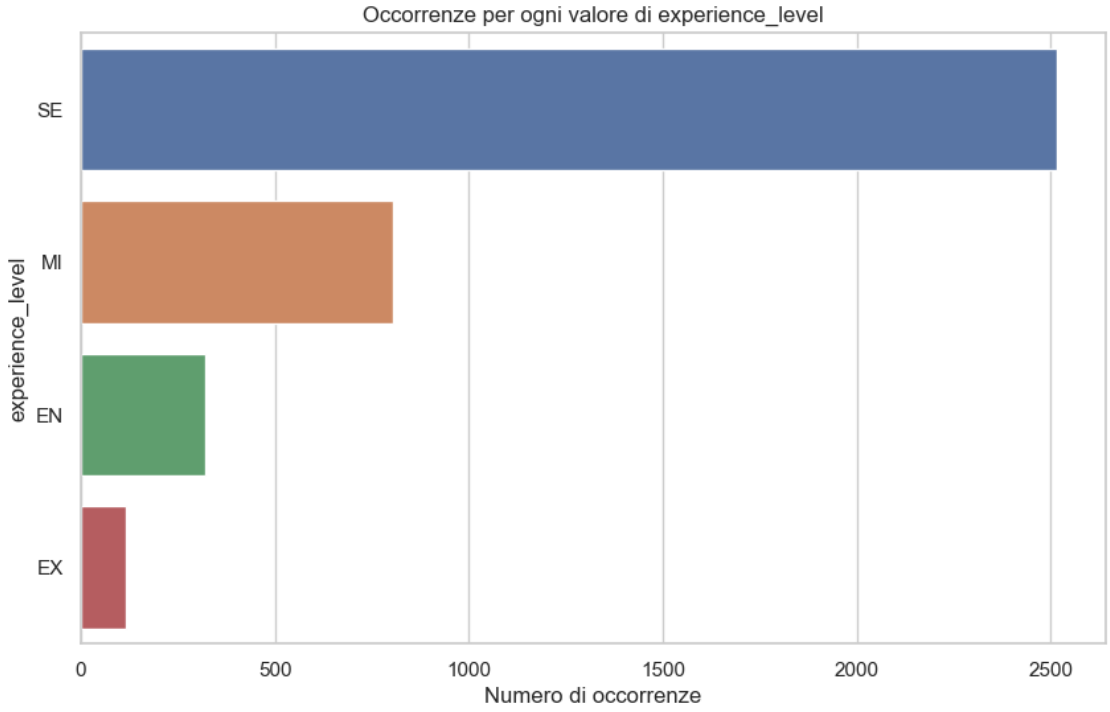
```
Numero di occorrenze per ogni valore di salary_currency:
USD    3224
EUR     236
GBP     161
INR      60
CAD      25
AUD       9
SGD       6
BRL       6
PLN       5
CHF       4
HUF       3
DKK       3
JPY       3
TRY       3
THB       2
ILS       1
HKD       1
CZK       1
MXN       1
CLP       1
Name: salary_currency, dtype: int64

Numero di occorrenze per ogni valore di salary_in_usd:
100000    99
150000    98
120000    91
160000    84
130000    82
          ..
234100     1
223800     1
172100     1
232200     1
94665      1
Name: salary_in_usd, Length: 1035, dtype: int64

Numero di occorrenze per ogni valore di employee_residence:
US    3004
GB     167
CA      85
ES      80
IN      71
      …
BA       1
AM       1
CY       1
KW       1
```

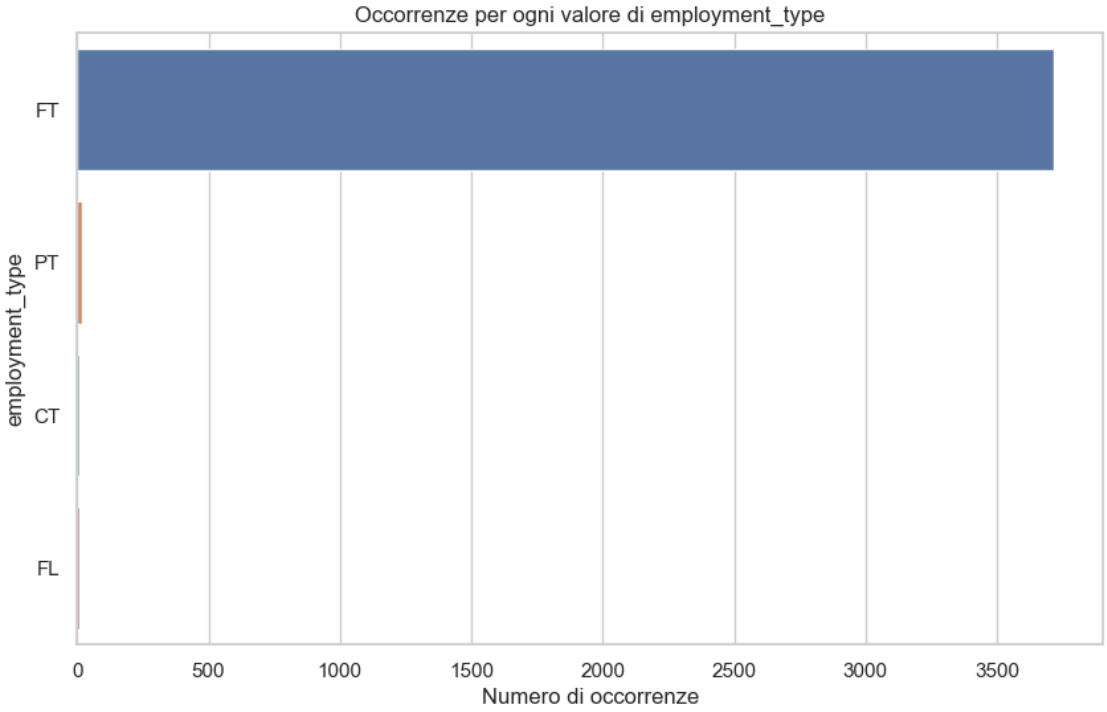
```
MT       1
Name: employee_residence, Length: 78, dtype: int64

Numero di occorrenze per ogni valore di remote_ratio:
0      1923
100    1643
50      189
Name: remote_ratio, dtype: int64

Numero di occorrenze per ogni valore di company_location:
US    3040
GB     172
CA      87
ES      77
IN      58
      …
MK       1
BS       1
IR       1
CR       1
MT       1
Name: company_location, Length: 72, dtype: int64

Numero di occorrenze per ogni valore di company_size:
M    3153
L     454
S     148
Name: company_size, dtype: int64
```

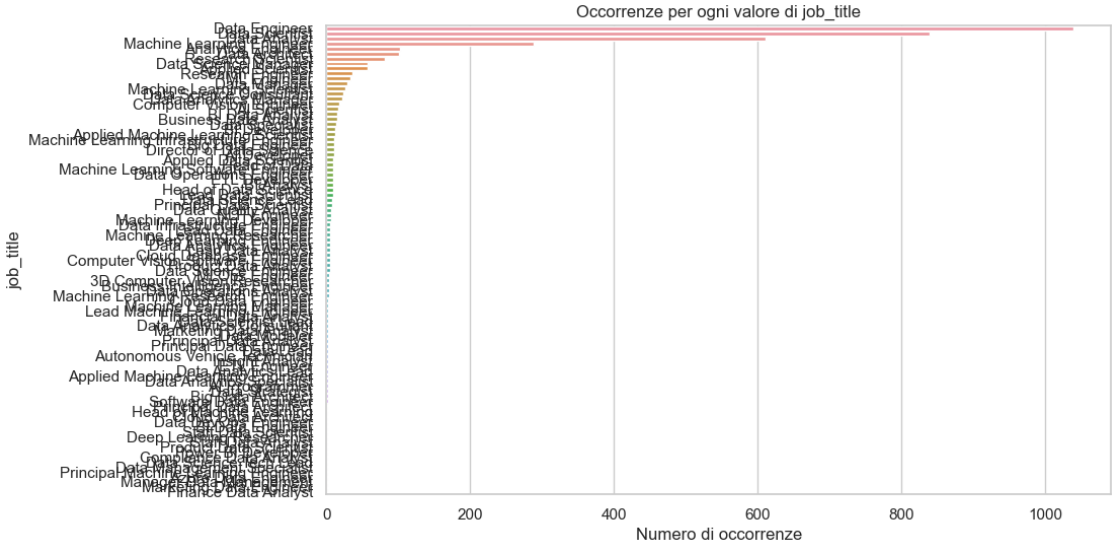
[ ]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Impostazione dello stile di visualizzazione per i grafici
sns.set(style="whitegrid")

# Definizione delle feature per le quali si vuole fare il grafico delle␣
 ↪occorrenze
features = ["work_year", "experience_level", "employment_type", "job_title",
            "salary_currency", "employee_residence", "remote_ratio",
            "company_location", "company_size"]

# Creazione dei grafici per ogni feature
for feature in features:
    plt.figure(figsize=(10, 6))  # Imposta le dimensioni del grafico
    sns.countplot(y=feature, data=dataset, order = dataset[feature].
 ↪value_counts().index)  # Crea il grafico a barre
    plt.title(f'Occorrenze per ogni valore di {feature}')  # Titolo del grafico
```

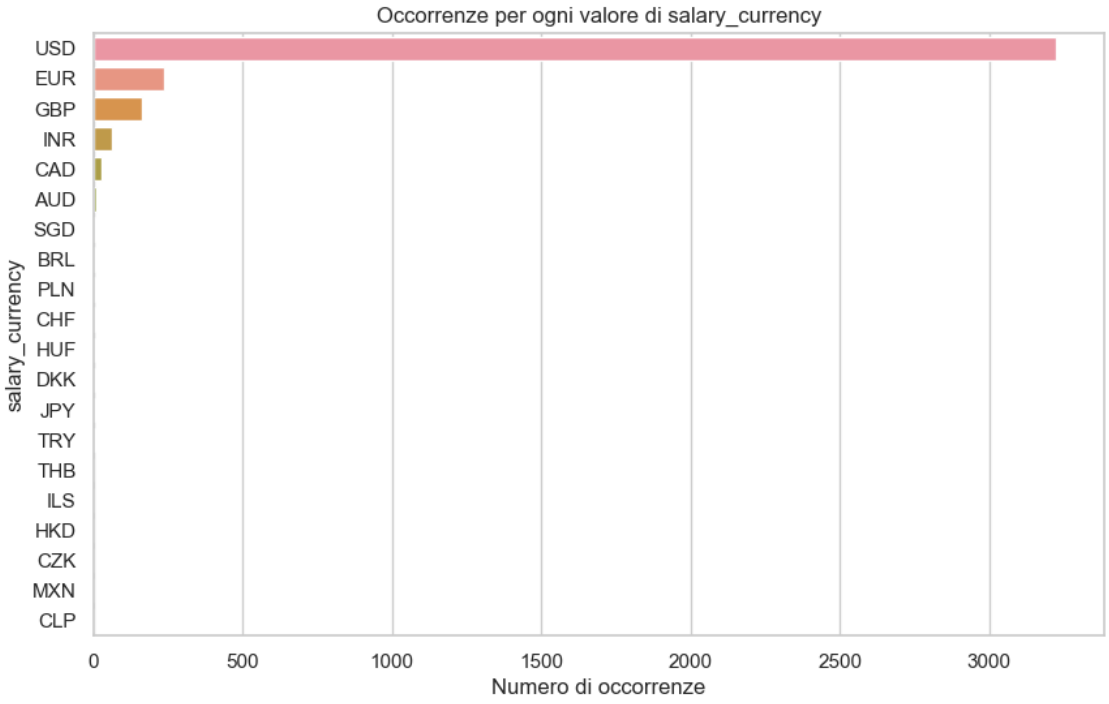
```
    plt.xlabel('Numero di occorrenze')  # Etichetta asse x
    plt.ylabel(feature)  # Etichetta asse y
    plt.show()  # Mostra il grafico
```

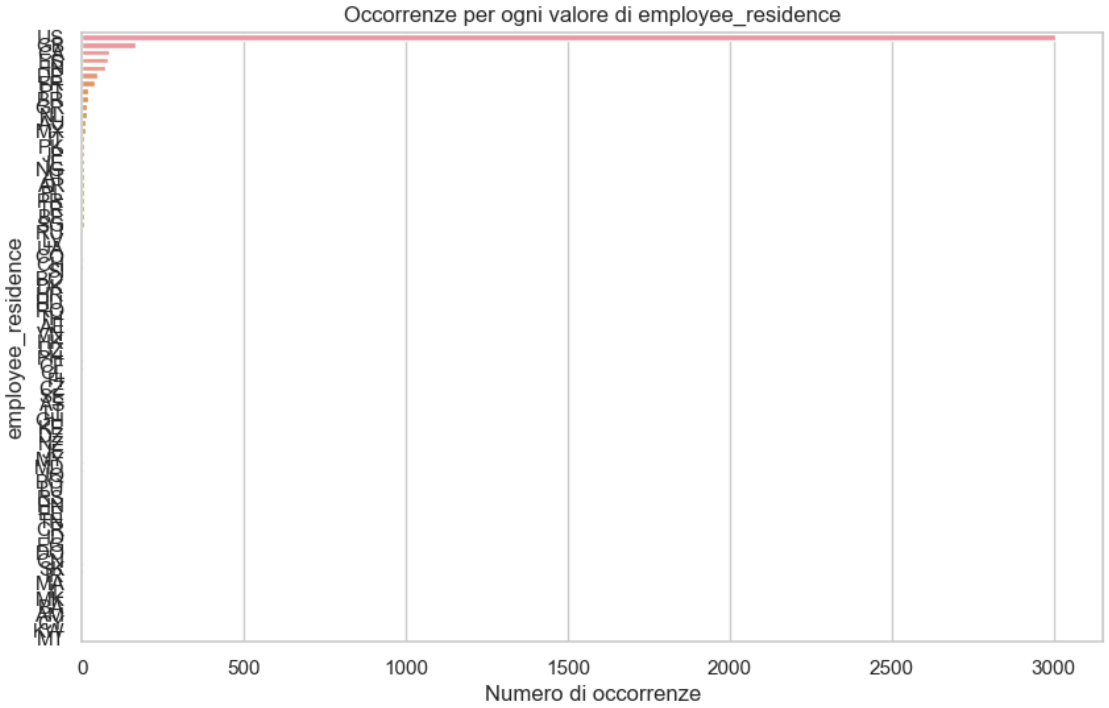


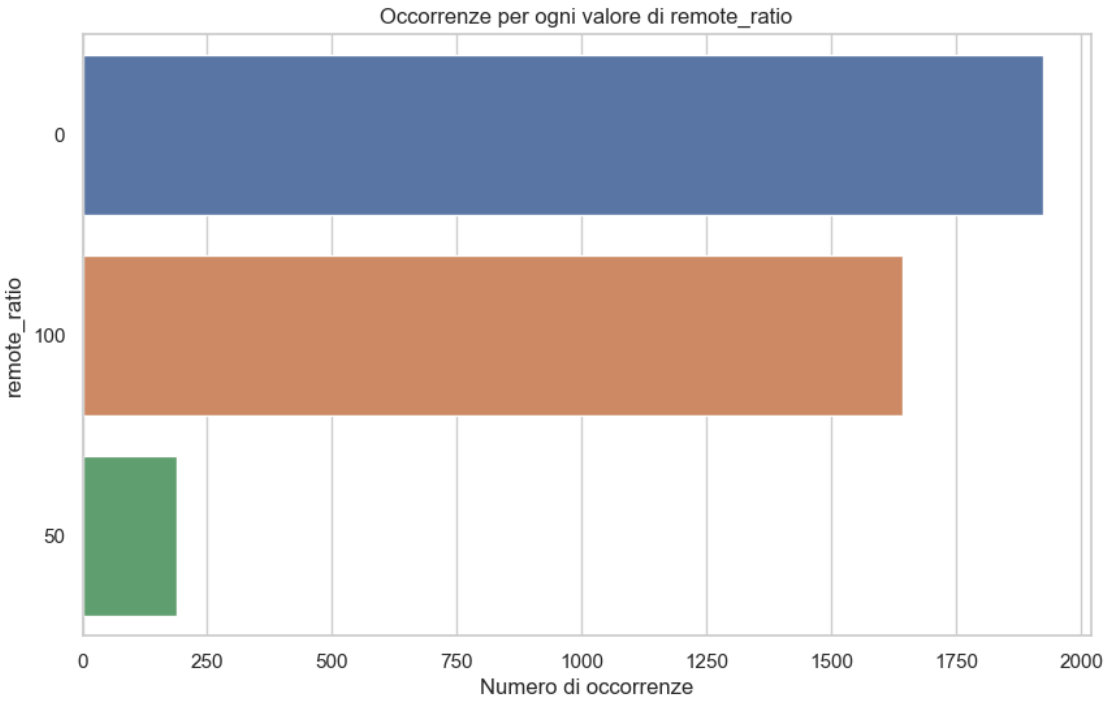


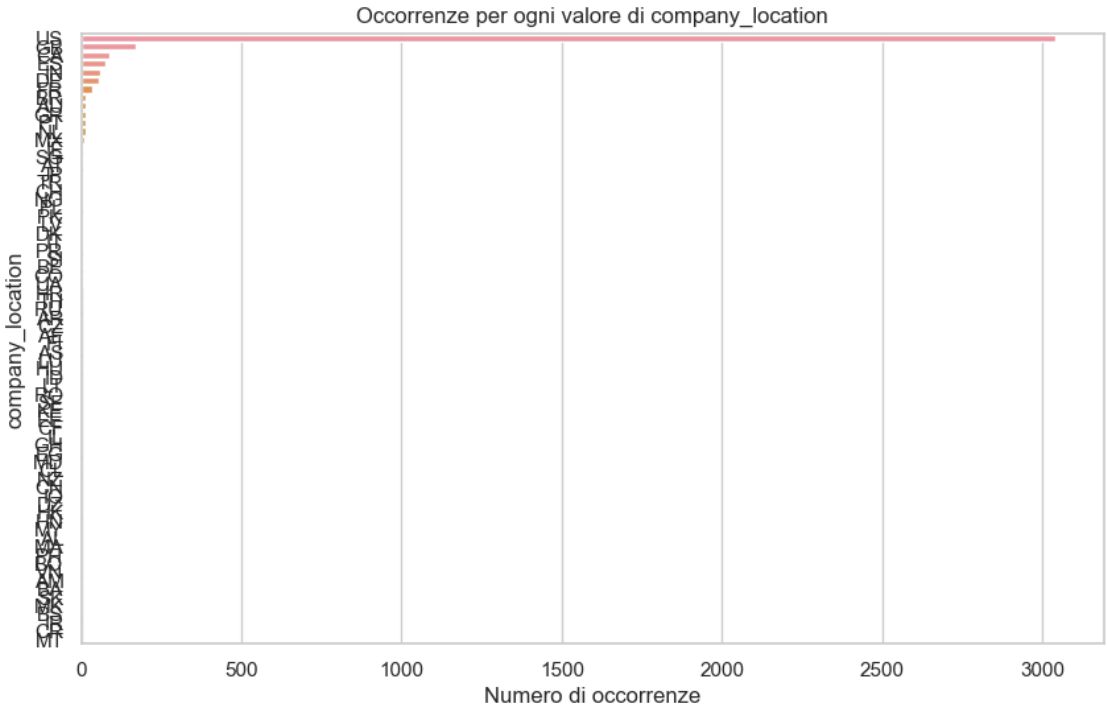

Occorrenze per ogni valore di employment_type
employment_type
FT
PT
CT
FL
0
500
1000
1500
2000
2500
3000
3500
Numero di occorrenze

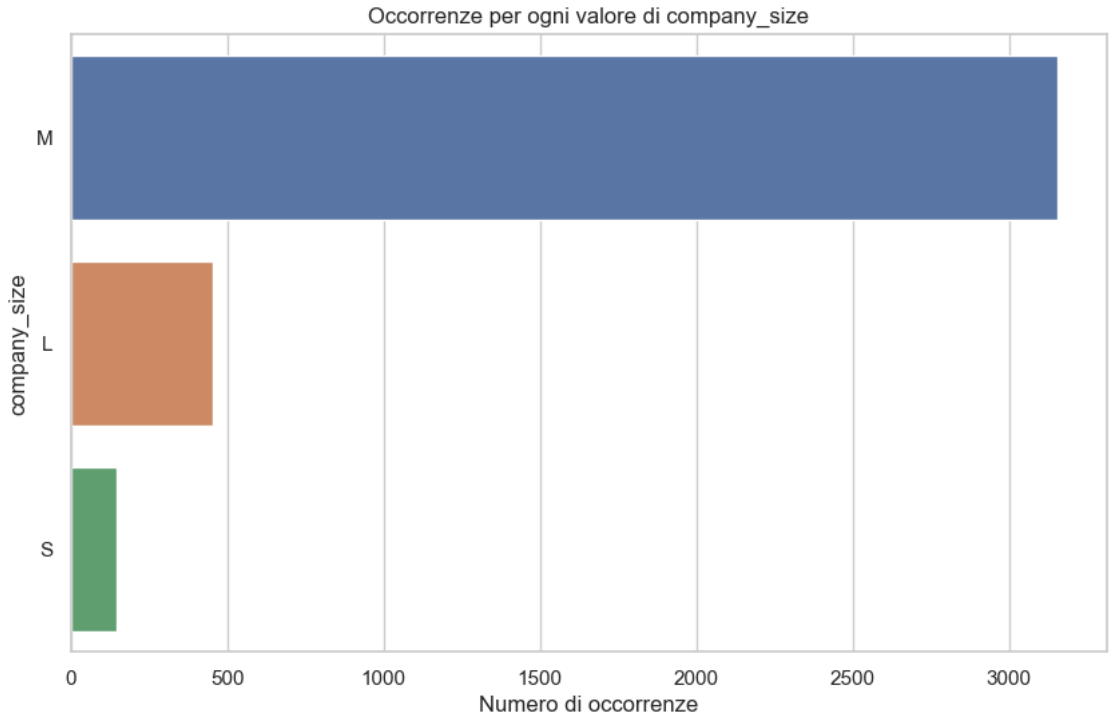

Occorrenze per ogni valore di job_title
job_title
0
200
400
600
800
1000
Numero di occorrenze


Occorrenze per ogni valore di salary_currency
salary_currency
USD
EUR
GBP
INR
CAD
AUD
SGD
BRL
PLN
CHF
HUF
DKK
JPY
TRY
THB
ILS
HKD
CZK
MXN
CLP
0
500
1000
1500
2000
2500
3000
Numero di occorrenze


Occorrenze per ogni valore di employee_residence
employee_residence
0
500
1000
1500
2000
2500
3000
Numero di occorrenze


Occorrenze per ogni valore di remote_ratio
remote_ratio
0
100
50
0
250
500
750
1000
1250
1500
1750
2000
Numero di occorrenze


Occorrenze per ogni valore di company_location
company_location
0
500
1000
1500
2000
2500
3000
Numero di occorrenze



[ ]:
```
import pandas as pd

valori_maggiori_dataset = dataset['salary'].nlargest(50)

valori_maggiori_dataset
```

[ ]:
```
3669    30400000
3574    11000000
3646    11000000
3475     8500000
2966     7500000
3476     7000000
3754     7000000
2358     6600000
3192     6000000
1462     5000000
3639     4900000
1868     4460000
3682     4450000
3061     4200000
1738     4000000
3649     4000000
2655     3000000
```

```
3489     3000000
3659     3000000
1946     2800000
1918     2500000
3678     2500000
3075     2400000
3423     2400000
3567     2250000
3581     2200000
3589     2100000
2786     1800000
3650     1799997
156      1700000
3666     1672000
41       1650000
3605     1600000
528      1500000
738      1500000
3734     1450000
1549     1440000
1739     1440000
217      1400000
735      1400000
3422     1400000
3426     1400000
1260     1350000
3729     1335000
988      1300000
1596     1250000
2032     1250000
3640     1250000
3644     1200000
1810     1125000
Name: salary, dtype: int64
```

1.3 FASE 3: MODIFICA DEL DATASET (CON I GRAFICI)

1) VOGLIAMO MODIFICARE IL DATASET CONSIDERANDO SOLO TRE FEATURES E CON TUTTI I SALARI IN DOLLARI
2) ELIMINARE LE FEATURE INUTILI AL NOSTRO ALGORITMO FINALE
3) SALVARE SOVRASCRIVENDO IL DATASET
4) STAMPARE IL NUOVO DATASET PER VERIFICARE SE LE OPERAZIONE FATTE PRECEDENENTE HANNO AVUTO UN SEGUITO POSITIVO

TUTTE LE MODIFICHE VENGONO FATTE SU UN DATASET CLONE, IN MODO POI DA POTERLO COMPARARE CON L'ORIGINALE

[ ]:
```
job_titles = ['Data Scientist', 'Machine Learning Engineer', 'Data Analyst',␣
 ↪'Data Engineer', 'Data Architect', 'Business Intelligence Engineer', 'Data␣
 ↪Strategist', 'Data Quality Analyst', 'Data Science Manager', 'Data␣
 ↪Operations Engineer']
print(len(job_titles))
dataset_ridotto = dataset[dataset['job_title'].isin(job_titles)]
dataset_ridotto["job_title"].unique() # Controllare che l'unico valore
```

```
10
```

[ ]:
```
array(['Data Scientist', 'Data Analyst', 'Business Intelligence Engineer',
       'Machine Learning Engineer', 'Data Strategist', 'Data Engineer',
       'Data Quality Analyst', 'Data Architect', 'Data Science Manager',
       'Data Operations Engineer'], dtype=object)
```

[ ]:
```
print("I valori di job_title sono:")
print(dataset_ridotto["job_title"].unique())
```

```
I valori di job_title sono:
['Data Scientist' 'Data Analyst' 'Business Intelligence Engineer'
 'Machine Learning Engineer' 'Data Strategist' 'Data Engineer'
 'Data Quality Analyst' 'Data Architect' 'Data Science Manager'
 'Data Operations Engineer']
```

[ ]:
```
dataset_ridotto=dataset[dataset["salary_currency"] == "USD"] # Filtrare le␣
 ↪righe (istanze) del dataset in cui i valori di salary currency è "USD"
dataset_ridotto["salary_currency"].unique() # Controllare che l'unico valore in␣
 ↪salary currency sia "USD"
```

[ ]:
```
array(['USD'], dtype=object)
```

[ ]:
```
print("I valori di salary_currency sono:")
print(dataset_ridotto["salary_currency"].unique())
```

```
I valori di salary_currency sono:
['USD']
```

[ ]:
```
dataset_ridotto=dataset[dataset["company_location"] == "US"] # Filtrare le␣
 ↪righe (istanze) del dataset in cui i valori di company location è "US"
dataset_ridotto["company_location"].unique() # Controllare che l'unico valore␣
 ↪in company location sia "US"
```

[ ]:
```
array(['US'], dtype=object)
```

[ ]:
```
print("I valori di company_location sono:")
print(dataset_ridotto["company_location"].unique())
```

```
I valori di company_location sono:
['US']
```

```
[ ]: dataset_ridotto=dataset_ridotto[dataset_ridotto["work_year"] == 2023] #␣
      ↪Filtrare le righe (istanze) del dataset
     dataset_ridotto["work_year"].unique() # Controllare che l'unico valore
```

```
[ ]: array([2023], dtype=int64)
```

```
[ ]: print("I valori di work_year sono:")
     print(dataset_ridotto["work_year"].unique())
```

```
I valori di work_year sono:
[2023]
```

```
[ ]: dataset=dataset[dataset["work_year"] == 2023] # Filtrare le righe (istanze) del␣
      ↪dataset in cui i valori di work_year è "2023"
     dataset["work_year"].unique() # Controllare che l'unico valore in work_year è␣
      ↪"2023"
```

```
[ ]: array([2023], dtype=int64)
```

```
[ ]: print("I valori di work_year sono:")
     print(dataset["work_year"].unique())
```

```
I valori di work_year sono:
[2023]
```

```
[ ]: dataset_ridotto =␣
      ↪dataset_ridotto[["experience_level","job_title","salary","company_location"]]␣
      ↪# Filtrare solo le features scelte e il target (salary). Le altre features␣
      ↪non scritte veranno eliminate
     dataset_ridotto
```

```
[ ]:      experience_level                job_title  salary company_location
     1                  MI               ML Engineer   30000               US
     2                  MI               ML Engineer   25500               US
     5                  SE         Applied Scientist  222200               US
     6                  SE         Applied Scientist  136000               US
     9                  SE            Data Scientist  147100               US
     …                 …                          …     …                …
     1815               SE  Machine Learning Engineer  134500               US
     1817               MI            Data Scientist  130000               US
     1818               MI            Data Scientist   90000               US
     1819               EN             Data Engineer  160000               US
     1820               EN             Data Engineer  135000               US

     [1570 rows x 4 columns]
```

```
[ ]:
```

```
dataset = dataset[["experience_level","job_title","salary","company_location"]]␣
 ↪# Filtrare solo le features scelte e il target (salary). Le altre features␣
 ↪non scritte veranno eliminate
dataset
```

[ ]:

	experience_level	job_title	salary	company_location
0	SE	Principal Data Scientist	80000	ES
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
3	SE	Data Scientist	175000	CA
4	SE	Data Scientist	120000	CA
…	…	…	…	…
1815	SE	Machine Learning Engineer	134500	US
1817	MI	Data Scientist	130000	US
1818	MI	Data Scientist	90000	US
1819	EN	Data Engineer	160000	US
1820	EN	Data Engineer	135000	US

[1785 rows x 4 columns]

[ ]:
```
dataset.duplicated().sum()
```

[ ]: 722

[ ]:
```
dataset = dataset.drop_duplicates()
```

[ ]:
```
dataset.duplicated().sum()
```

[ ]: 0

[ ]:
```
dataset_ridotto.duplicated().sum()
```

[ ]: 703

[ ]:
```
dataset_ridotto = dataset_ridotto.drop_duplicates()
```

[ ]:
```
dataset_ridotto.duplicated().sum()
```

[ ]: 0

[ ]:
```
dataset
```

[ ]:

	experience_level	job_title	salary	company_location
0	SE	Principal Data Scientist	80000	ES
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
3	SE	Data Scientist	175000	CA
4	SE	Data Scientist	120000	CA

```
…                …                          …      …                …
1809                SE              Data Engineer  182000                US
1814                SE  Machine Learning Engineer  261500                US
1815                SE  Machine Learning Engineer  134500                US
1817                MI             Data Scientist  130000                US
1818                MI             Data Scientist   90000                US

[1063 rows x 4 columns]
```

[ ]:
```
dataset_ridotto
```

[ ]:
```
     experience_level                job_title  salary company_location
1                   MI               ML Engineer   30000                US
2                   MI               ML Engineer   25500                US
5                   SE          Applied Scientist  222200                US
6                   SE          Applied Scientist  136000                US
9                   SE             Data Scientist  147100                US
…                …                          …      …                …
1809                SE              Data Engineer  182000                US
1814                SE  Machine Learning Engineer  261500                US
1815                SE  Machine Learning Engineer  134500                US
1817                MI             Data Scientist  130000                US
1818                MI             Data Scientist   90000                US

[867 rows x 4 columns]
```

1.4 FASE 4: LE DISTRIBUZIONI E I GRAFICI SULLE MODIFICHE DEL DATASET RISPETTO AL DATASET ORIGINALE (CON I GRAFICI)

1) CONFRONTIAMO LE DISTRIBUZIONE DEI TITOLI DI LAVORI “MONDIALE” VS CON QUELLA AMERICANA

[ ]:
```
from matplotlib import pyplot as plt

persone_totali = len(dataset)

# Calcolare le percentuali dei titoli di lavoro mondiali rispetto ad una␣
 ↪singola categoria di lavoro

# Calcolare percentuali di "Data Scientist" mondiali

DataScientist_mondiali = dataset[dataset["job_title"]=="Data Scientist"]
numero_DataScientist_mondiali = len(DataScientist_mondiali)
percentuale_DataScientist_mondiali = numero_DataScientist_mondiali/
 ↪persone_totali*100

# Calcolare percentuali di "Machine Learning Engineer" mondiali
```

```
Machine_Learning_Engineer_mondiali = dataset[dataset["job_title"]=="Machine␣
 ↪Learning Engineer"]
numero_Machine_Learning_Engineer_mondiali =␣
 ↪len(Machine_Learning_Engineer_mondiali)
percentuale_Machine_Learning_Engineer_mondiali =␣
 ↪numero_Machine_Learning_Engineer_mondiali/persone_totali*100

# Calcolare percentuali di "Data Analyst" mondiali

Data_Analyst_mondiali = dataset[dataset["job_title"]=="Data Analyst"]
numero_Data_Analyst_mondiali = len(Data_Analyst_mondiali)
percentuale_Data_Analyst_mondiali = numero_Data_Analyst_mondiali/
 ↪persone_totali*100

# Calcolare percentuali di "Data Engineer" mondiali

Data_Engineer_mondiali = dataset[dataset["job_title"]=="Data Engineer"]
numero_Data_Engineer_mondiali = len(Data_Engineer_mondiali)
percentuale_Data_Engineer_mondiali = numero_Data_Engineer_mondiali/
 ↪persone_totali*100

# Calcolare percentuali di "Data Architect" mondiali

Data_Architect_mondiali = dataset[dataset["job_title"]=="Data Architect"]
numero_Data_Architect_mondiali = len(Data_Architect_mondiali)
percentuale_Data_Architect_mondiali = numero_Data_Architect_mondiali/
 ↪persone_totali*100

# Calcolare percentuali di "Business Intelligence Engineer" mondiali

Business_Intelligence_Engineer_mondiali =␣
 ↪dataset[dataset["job_title"]=="Business Intelligence Engineer"]
numero_Business_Intelligence_Engineer_mondiali =␣
 ↪len(Business_Intelligence_Engineer_mondiali)
percentuale_Business_Intelligence_Engineer_mondiali =␣
 ↪numero_Business_Intelligence_Engineer_mondiali/persone_totali*100

# Calcolare percentuali di "Data Strategist" mondiali

Data_Strategist_mondiali = dataset[dataset["job_title"]=="Data Strategist"]
numero_Data_Strategist_mondiali = len(Data_Strategist_mondiali)
percentuale_Data_Strategist_mondiali = numero_Data_Strategist_mondiali/
 ↪persone_totali*100

# Calcolare percentuali di "Data Quality Analyst" mondiali
```

```
Data_Quality_Analyst_mondiali = dataset[dataset["job_title"]=="Data Quality␣
 ↪Analyst"]
numero_Data_Quality_Analyst_mondiali = len(Data_Quality_Analyst_mondiali)
percentuale_Data_Quality_Analyst_mondiali =␣
 ↪numero_Data_Quality_Analyst_mondiali/persone_totali*100

# Calcolare percentuali di "Data Science Manager" mondiali

Data_Science_Manager_mondiali = dataset[dataset["job_title"]=="Data Science␣
 ↪Manager"]
numero_Data_Science_Manager_mondiali = len(Data_Science_Manager_mondiali)
percentuale_Data_Science_Manager_mondiali =␣
 ↪numero_Data_Science_Manager_mondiali/persone_totali*100

# Calcolare percentuali di "Data Operations Engineer" mondiali

Data_Operations_Engineer_mondiali = dataset[dataset["job_title"]=="Data␣
 ↪Operations Engineer"]
numero_Data_Operations_Engineer_mondiali =␣
 ↪len(Data_Operations_Engineer_mondiali)
percentuale_Data_Operations_Engineer_mondiali =␣
 ↪numero_Data_Operations_Engineer_mondiali/persone_totali*100
```

[ ]:
```
from matplotlib import pyplot as plt

persone_totali = len(dataset_ridotto)

# Calcolare le percentuali dei titoli di lavoro americani rispetto ad una␣
 ↪singola categoria di lavoro

# Calcolare percentuali di "Data Scientist" americani

DataScientist_americani = dataset_ridotto[dataset_ridotto["job_title"]=="Data␣
 ↪Scientist"]
numero_DataScientist_americani = len(DataScientist_americani)
percentuale_DataScientist_americani = numero_DataScientist_americani/
 ↪persone_totali*100

# Calcolare percentuali di "Machine Learning Engineer" americani

Machine_Learning_Engineer_americani =␣
 ↪dataset_ridotto[dataset_ridotto["job_title"]=="Machine Learning Engineer"]
numero_Machine_Learning_Engineer_americani =␣
 ↪len(Machine_Learning_Engineer_americani)
```

```
percentuale_Machine_Learning_Engineer_americani =␣
 ↪numero_Machine_Learning_Engineer_americani/persone_totali*100

# Calcolare percentuali di "Data Analyst" americani

Data_Analyst_americani = dataset_ridotto[dataset_ridotto["job_title"]=="Data␣
 ↪Analyst"]
numero_Data_Analyst_americani = len(Data_Analyst_americani)
percentuale_Data_Analyst_americani = numero_Data_Analyst_americani/
 ↪persone_totali*100

# Calcolare percentuali di "Data Engineer" americani

Data_Engineer_americani = dataset_ridotto[dataset_ridotto["job_title"]=="Data␣
 ↪Engineer"]
numero_Data_Engineer_americani = len(Data_Engineer_americani)
percentuale_Data_Engineer_americani = numero_Data_Engineer_americani/
 ↪persone_totali*100

# Calcolare percentuali di "Data Architect" americani

Data_Architect_americani = dataset_ridotto[dataset_ridotto["job_title"]=="Data␣
 ↪Architect"]
numero_Data_Architect_americani = len(Data_Architect_americani)
percentuale_Data_Architect_americani = numero_Data_Architect_americani/
 ↪persone_totali*100

# Calcolare percentuali di "Business Intelligence Engineer" americani

Business_Intelligence_Engineer_americani =␣
 ↪dataset_ridotto[dataset_ridotto["job_title"]=="Business Intelligence␣
 ↪Engineer"]
numero_Business_Intelligence_Engineer_americani =␣
 ↪len(Business_Intelligence_Engineer_americani)
percentuale_Business_Intelligence_Engineer_americani =␣
 ↪numero_Business_Intelligence_Engineer_americani/persone_totali*100

# Calcolare percentuali di "Data Strategist" americani

Data_Strategist_americani = dataset_ridotto[dataset_ridotto["job_title"]=="Data␣
 ↪Strategist"]
numero_Data_Strategist_americani = len(Data_Strategist_americani)
percentuale_Data_Strategist_americani = numero_Data_Strategist_americani/
 ↪persone_totali*100

# Calcolare percentuali di "Data Quality Analyst" americani
```

```
Data_Quality_Analyst_americani =␣
 ↪dataset_ridotto[dataset_ridotto["job_title"]=="Data Quality Analyst"]
numero_Data_Quality_Analyst_americani = len(Data_Quality_Analyst_americani)
percentuale_Data_Quality_Analyst_americani =␣
 ↪numero_Data_Quality_Analyst_americani/persone_totali*100

# Calcolare percentuali di "Data Science Manager" americani

Data_Science_Manager_americani =␣
 ↪dataset_ridotto[dataset_ridotto["job_title"]=="Data Science Manager"]
numero_Data_Science_Manager_americani = len(Data_Science_Manager_americani)
percentuale_Data_Science_Manager_americani =␣
 ↪numero_Data_Science_Manager_americani/persone_totali*100

# Calcolare percentuali di "Data Operations Engineer" americani

Data_Operations_Engineer_americani =␣
 ↪dataset_ridotto[dataset_ridotto["job_title"]=="Data Operations Engineer"]
numero_Data_Operations_Engineer_americani =␣
 ↪len(Data_Operations_Engineer_americani)
percentuale_Data_Operations_Engineer_americani =␣
 ↪numero_Data_Operations_Engineer_americani/persone_totali*100
```

[ ]:
```
print("Le percentuali mondiali di \"Data Scientist\" sono:")
print(percentuale_DataScientist_mondiali)
print("Le percentuali mondiali di \"Machine Learning Engineer\" sono:")
print(percentuale_Machine_Learning_Engineer_mondiali)
print("Le percentuali mondiali di \"Data Analyst\" sono:")
print(percentuale_Data_Analyst_mondiali)
print("Le percentuali mondiali di \"Data Engineer\" sono:")
print(percentuale_Data_Engineer_mondiali)
print("Le percentuali mondiali di \"Data Architect\" sono:")
print(percentuale_Data_Architect_mondiali)
print("Le percentuali mondiali di \"Business Intelligence Engineer\" sono:")
print(percentuale_Business_Intelligence_Engineer_mondiali)
print("Le percentuali mondiali di \"Data Strategist\" sono:")
print(percentuale_Data_Strategist_mondiali)
print("Le percentuali mondiali di \"Data Quality Analyst\" sono:")
print(percentuale_Data_Quality_Analyst_mondiali)
print("Le percentuali mondiali di \"Data Science Manager\" sono:")
print(percentuale_Data_Science_Manager_mondiali)
print("Le percentuali mondiali di \"Data Operations Engineer\" sono:")
print(percentuale_Data_Operations_Engineer_mondiali)
```

```
Le percentuali mondiali di "Data Scientist" sono:
19.285042333019756
```

```
Le percentuali mondiali di "Machine Learning Engineer" sono:
9.125117591721544
Le percentuali mondiali di "Data Analyst" sono:
15.61618062088429
Le percentuali mondiali di "Data Engineer" sono:
22.295390404515523
Le percentuali mondiali di "Data Architect" sono:
2.916274694261524
Le percentuali mondiali di "Business Intelligence Engineer" sono:
0.37629350893697083
Le percentuali mondiali di "Data Strategist" sono:
0.18814675446848542
Le percentuali mondiali di "Data Quality Analyst" sono:
0.4703668861712135
Le percentuali mondiali di "Data Science Manager" sono:
1.5051740357478833
Le percentuali mondiali di "Data Operations Engineer" sono:
0.18814675446848542
```

[ ]:
```
print("Le percentuali americane di \"Data Scientist\" sono:")
print(percentuale_DataScientist_americani)
print("Le percentuali americane di \"Machine Learning Engineer\" sono:")
print(percentuale_Machine_Learning_Engineer_americani)
print("Le percentuali americane di \"Data Analyst\" sono:")
print(percentuale_Data_Analyst_americani)
print("Le percentuali americane di \"Data Engineer\" sono:")
print(percentuale_Data_Engineer_americani)
print("Le percentuali americane di \"Data Architect\" sono:")
print(percentuale_Data_Architect_americani)
print("Le percentuali americane di \"Business Intelligence Engineer\" sono:")
print(percentuale_Business_Intelligence_Engineer_americani)
print("Le percentuali americane di \"Data Strategist\" sono:")
print(percentuale_Data_Strategist_americani)
print("Le percentuali americane di \"Data Quality Analyst\" sono:")
print(percentuale_Data_Quality_Analyst_americani)
print("Le percentuali americane di \"Data Science Manager\" sono:")
print(percentuale_Data_Science_Manager_americani)
print("Le percentuali americane di \"Data Operations Engineer\" sono:")
print(percentuale_Data_Operations_Engineer_americani)
```

```
Le percentuali americane di "Data Scientist" sono:
18.569780853517877
Le percentuali americane di "Machine Learning Engineer" sono:
8.535178777393309
Le percentuali americane di "Data Analyst" sono:
16.147635524798154
Le percentuali americane di "Data Engineer" sono:
24.22145328719723
```

```
Le percentuali americane di "Data Architect" sono:
3.3448673587081887
Le percentuali americane di "Business Intelligence Engineer" sono:
0.461361014994233
Le percentuali americane di "Data Strategist" sono:
0.0
Le percentuali americane di "Data Quality Analyst" sono:
0.461361014994233
Le percentuali americane di "Data Science Manager" sono:
1.6147635524798154
Le percentuali americane di "Data Operations Engineer" sono:
0.2306805074971165
```

[ ]:
```
# Calcolo percentuale totale mondiale
percentuale_totale_mondiale = percentuale_Data_Analyst_mondiali +␣
  ↪percentuale_Data_Engineer_mondiali + percentuale_DataScientist_mondiali +␣
  ↪percentuale_Data_Architect_mondiali +␣
  ↪percentuale_Data_Quality_Analyst_mondiali +␣
  ↪percentuale_Data_Science_Manager_mondiali +␣
  ↪percentuale_Data_Operations_Engineer_mondiali +␣
  ↪percentuale_Machine_Learning_Engineer_mondiali +␣
  ↪percentuale_Business_Intelligence_Engineer_mondiali +␣
  ↪percentuale_Data_Strategist_mondiali
print(f"La percentuale totale mondiale è pari a:␣
  ↪{int(percentuale_totale_mondiale)}%")
```

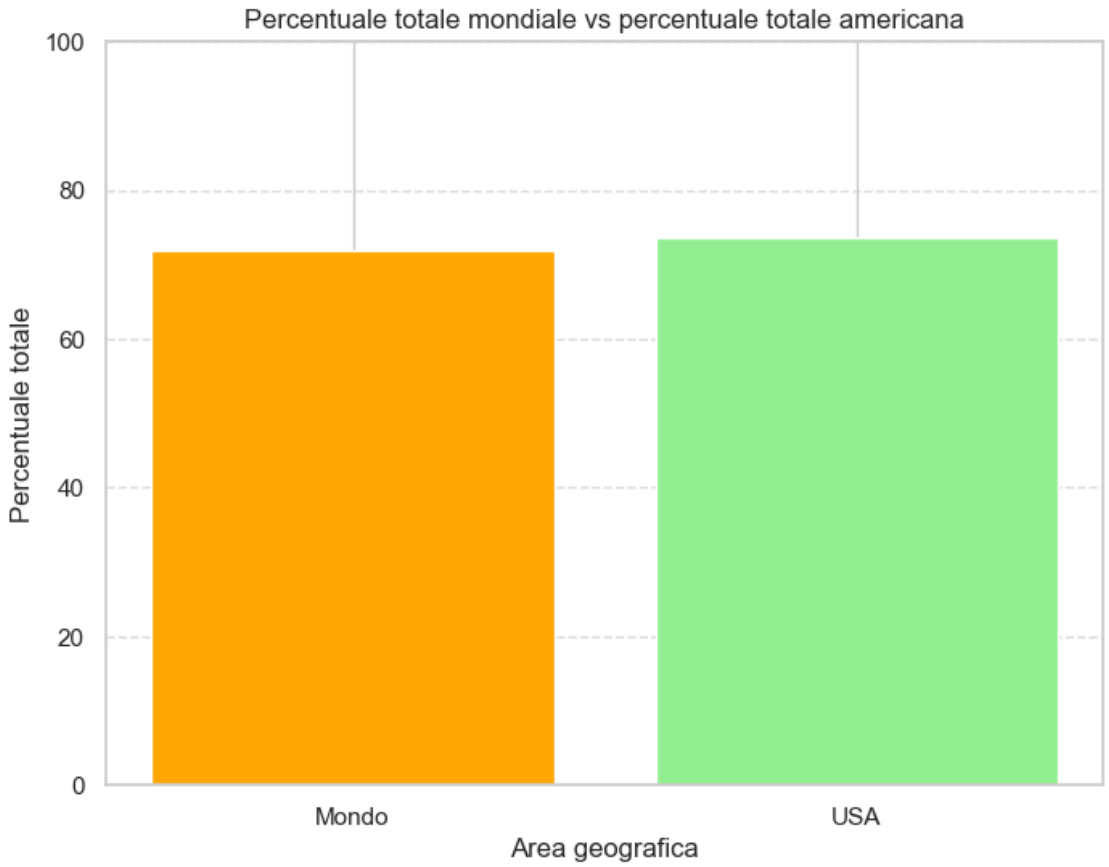
```
La percentuale totale mondiale è pari a: 71%
```

[ ]:
```
# Calcolo percentuale totale americana
percentuale_totale_americana = percentuale_Data_Analyst_americani +␣
  ↪percentuale_Data_Engineer_americani + percentuale_DataScientist_americani +␣
  ↪percentuale_Data_Architect_americani +␣
  ↪percentuale_Data_Quality_Analyst_americani +␣
  ↪percentuale_Data_Science_Manager_americani +␣
  ↪percentuale_Data_Operations_Engineer_americani +␣
  ↪percentuale_Machine_Learning_Engineer_americani +␣
  ↪percentuale_Business_Intelligence_Engineer_americani +␣
  ↪percentuale_Data_Strategist_americani
print(f"La percentuale totale americana è pari a:␣
  ↪{int(percentuale_totale_americana)}%")
```

```
La percentuale totale americana è pari a: 73%
```

[ ]:
```
import matplotlib.pyplot as plt

# Dati delle percentuali totali
percentuali_totali = [percentuale_totale_mondiale, percentuale_totale_americana]
etichette = ['Mondo', 'USA']
```

```
# Creazione del grafico a barre
plt.figure(figsize=(8, 6))
plt.bar(etichette, percentuali_totali, color=['orange', 'lightgreen'])
plt.xlabel('Area geografica')
plt.ylabel('Percentuale totale')
plt.title('Percentuale totale mondiale vs percentuale totale americana')
plt.ylim(0, 100)  # Impostazione del limite dell'asse y da 0 a 100
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Visualizzazione del grafico
plt.show()
```

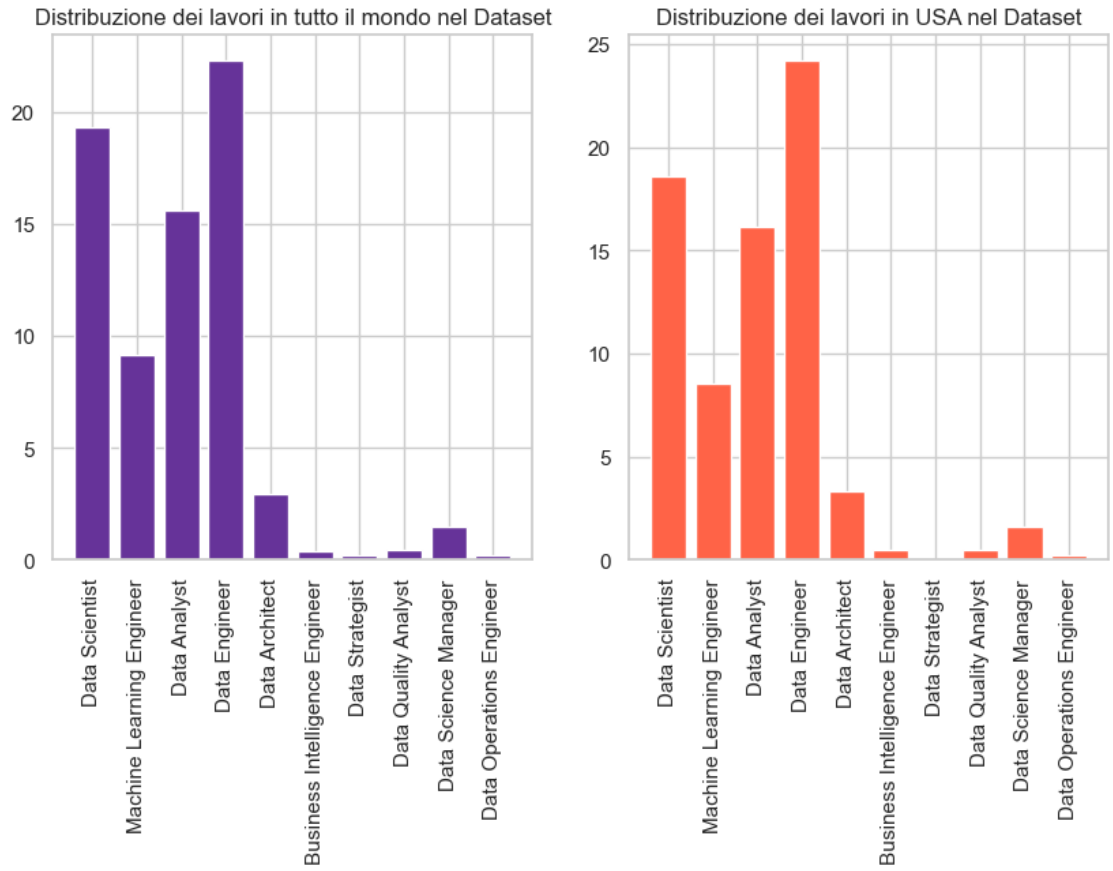


[ ]:
```
labels = job_titles
```

```
percentuali_mondiali = [percentuale_DataScientist_mondiali,␣
 ↪percentuale_Machine_Learning_Engineer_mondiali,␣
 ↪percentuale_Data_Analyst_mondiali, percentuale_Data_Engineer_mondiali,␣
 ↪percentuale_Data_Architect_mondiali,␣
 ↪percentuale_Business_Intelligence_Engineer_mondiali,␣
 ↪percentuale_Data_Strategist_mondiali,␣
 ↪percentuale_Data_Quality_Analyst_mondiali,␣
 ↪percentuale_Data_Science_Manager_mondiali,␣
 ↪percentuale_Data_Operations_Engineer_mondiali]
print(len(percentuali_mondiali))
plt.figure()
fig,axs = plt.subplots(1,2,figsize=(10,5))
axs[0].set_title("Distribuzione dei lavori in tutto il mondo nel Dataset")
axs[0].bar(labels,percentuali_mondiali, color="rebeccapurple")
axs[0].tick_params(axis='x',rotation=90)

percentuali_americani = [percentuale_DataScientist_americani,␣
 ↪percentuale_Machine_Learning_Engineer_americani,␣
 ↪percentuale_Data_Analyst_americani, percentuale_Data_Engineer_americani,␣
 ↪percentuale_Data_Architect_americani,␣
 ↪percentuale_Business_Intelligence_Engineer_americani,␣
 ↪percentuale_Data_Strategist_americani,␣
 ↪percentuale_Data_Quality_Analyst_americani,␣
 ↪percentuale_Data_Science_Manager_americani,␣
 ↪percentuale_Data_Operations_Engineer_americani]

axs[1].set_title("Distribuzione dei lavori in USA nel Dataset")
axs[1].bar(labels,percentuali_americani, color="tomato")
axs[1].tick_params(axis='x',rotation=90)

plt.show()
```

```
10
```

```
<Figure size 640x480 with 0 Axes>
```



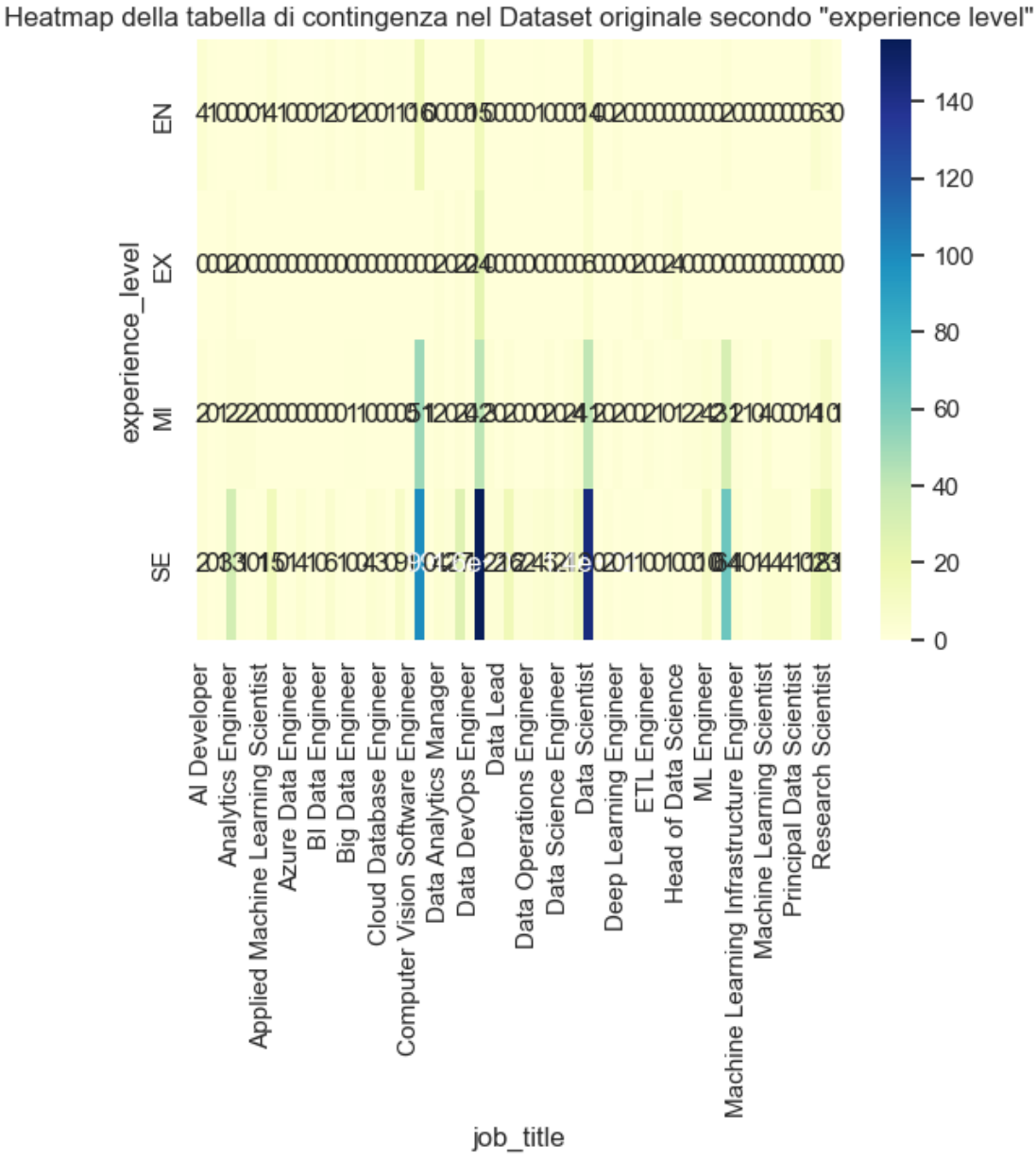
1.5 FASE 5: LE CORRELAZIONI TRA TUTTE LE FEATURES E IL SALARIO (CON IL CHI QUADRO)

1.5.1 PER IL DATASET ORIGINALE

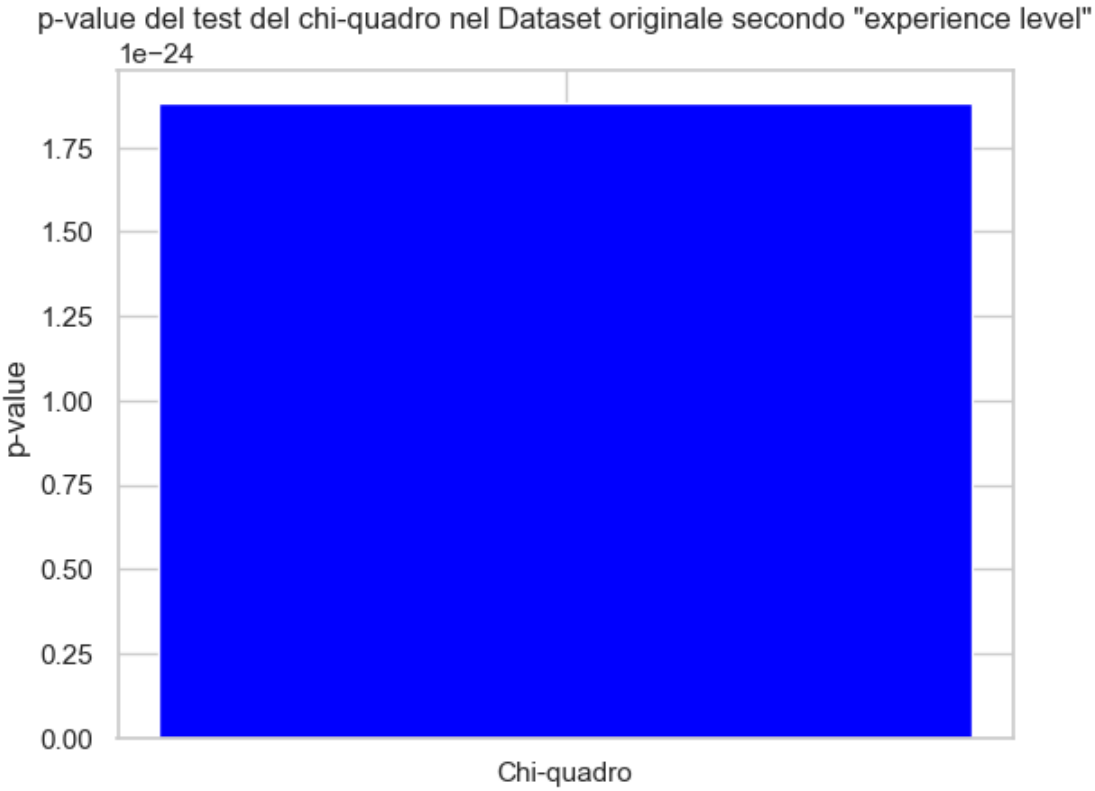
[ ]:
```
import pandas as pd
from scipy.stats import chi2_contingency

# Calcola la tabella di contingenza tra 'salary' e 'categoria'
contingency_table = pd.crosstab(dataset['experience_level'],␣
 ↪dataset['job_title'])

# Esegui il test del chi-quadro
chi2, p_value, dof, expected = chi2_contingency(contingency_table)

# Visualizza il p-value
print("P-value:", p_value)
```

```
P-value: 1.883138384719936e-24
```

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Utilizza seaborn per creare un heatmap della tabella di contingenza
sns.heatmap(contingency_table, annot=True, cmap="YlGnBu")

# Aggiungi titolo
plt.title('Heatmap della tabella di contingenza nel Dataset originale secondo␣
 ↪\"experience level\"')

# Mostra il grafico
plt.show()
```



[ ]:
```
import matplotlib.pyplot as plt

# Crea un barplot del p-value
plt.bar(['Chi-quadro'], [p_value], color='blue')

# Aggiungi titoli e label
plt.ylabel('p-value')
plt.title('p-value del test del chi-quadro nel Dataset originale secondo␣
 ↪\"experience level\"')

# Mostra il grafico
plt.show()
```

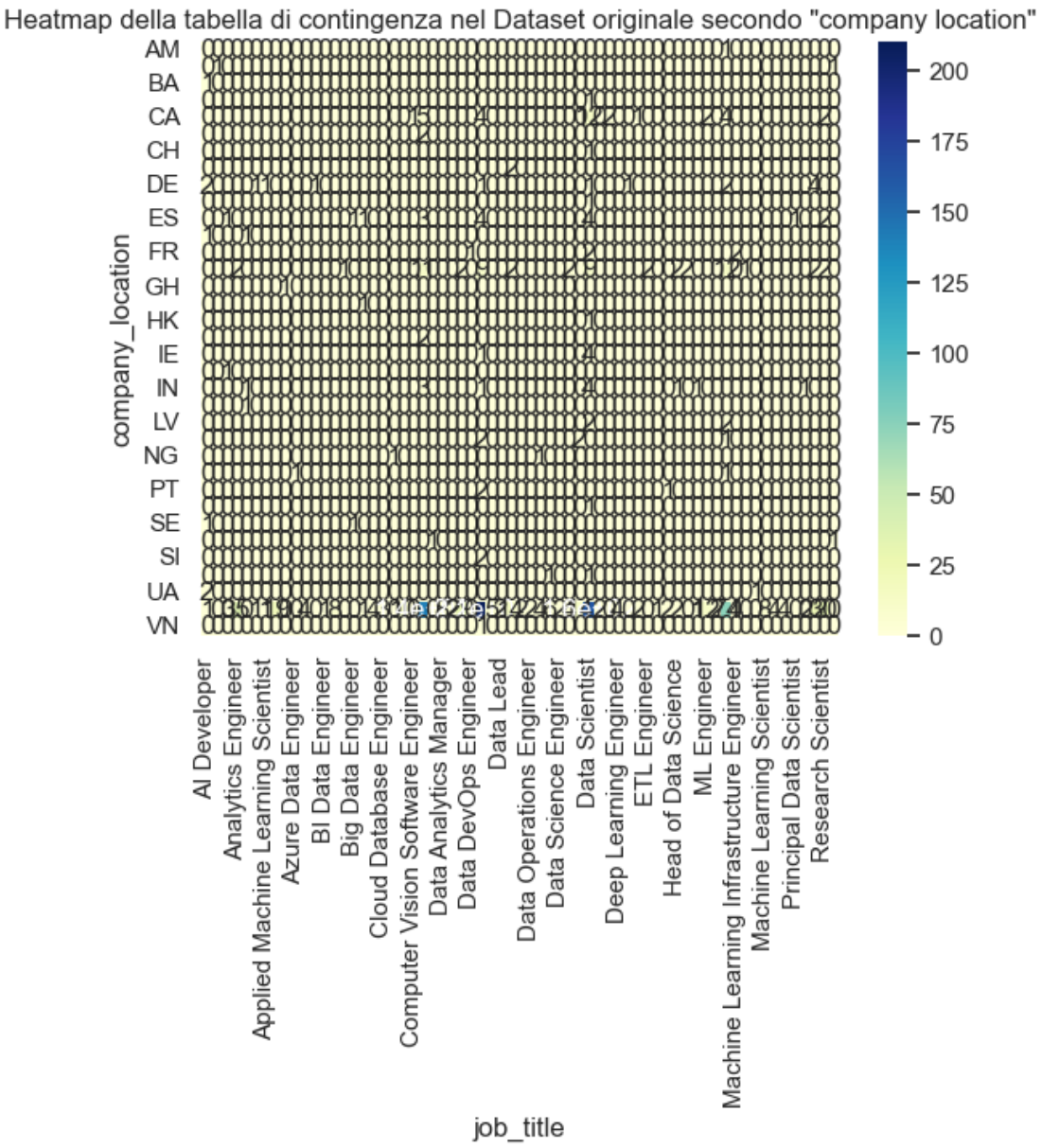


[ ]:
```
import pandas as pd
from scipy.stats import chi2_contingency

# Calcola la tabella di contingenza tra 'salary' e 'categoria'
```

```
contingency_table2 = pd.crosstab(dataset['company_location'],␣
 ↪dataset['job_title'])

# Esegui il test del chi-quadro
chi2, p_value2, dof, expected = chi2_contingency(contingency_table2)

# Visualizza il p-value
print("P-value:", p_value2)
```

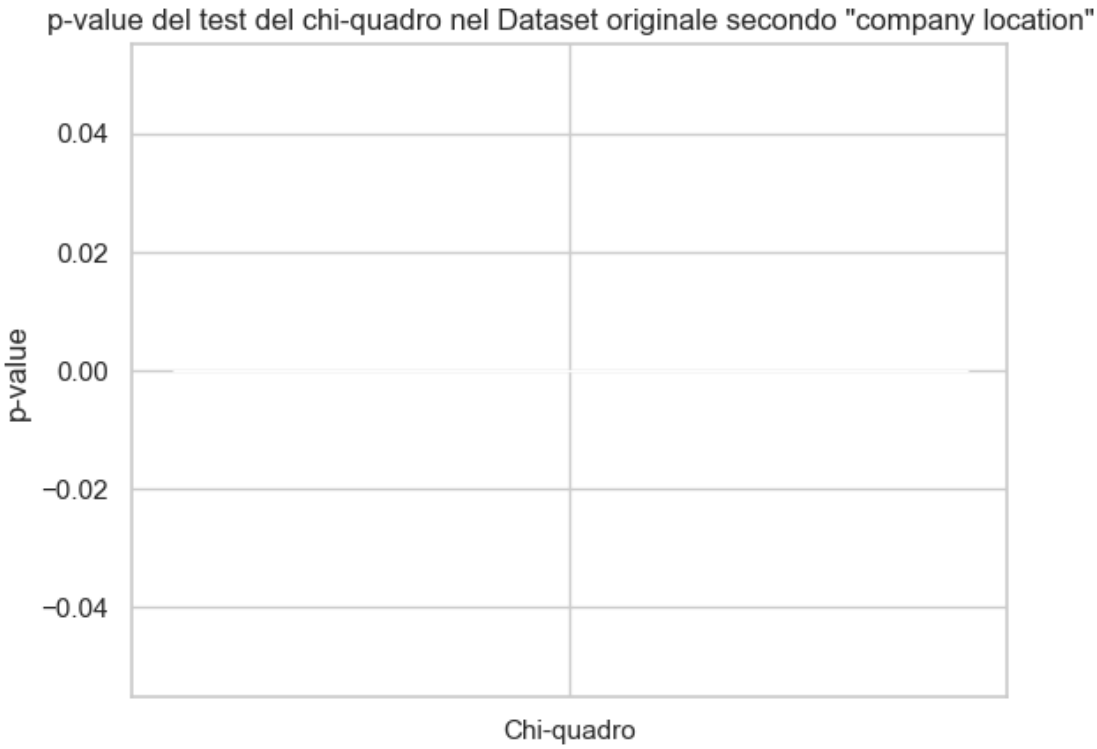
```
P-value: 0.0
```

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Utilizza seaborn per creare un heatmap della tabella di contingenza
sns.heatmap(contingency_table2, annot=True, cmap="YlGnBu")

# Aggiungi titolo
plt.title('Heatmap della tabella di contingenza nel Dataset originale secondo␣
 ↪\"company location\"')

# Mostra il grafico
plt.show()
```



[ ]:
```
import matplotlib.pyplot as plt

# Crea un barplot del p-value
plt.bar(['Chi-quadro'], [p_value2], color='pink')

# Aggiungi titoli e label
plt.ylabel('p-value')
plt.title('p-value del test del chi-quadro nel Dataset originale secondo␣
 ↪\"company location\"')
```

```
# Mostra il grafico
plt.show()
```



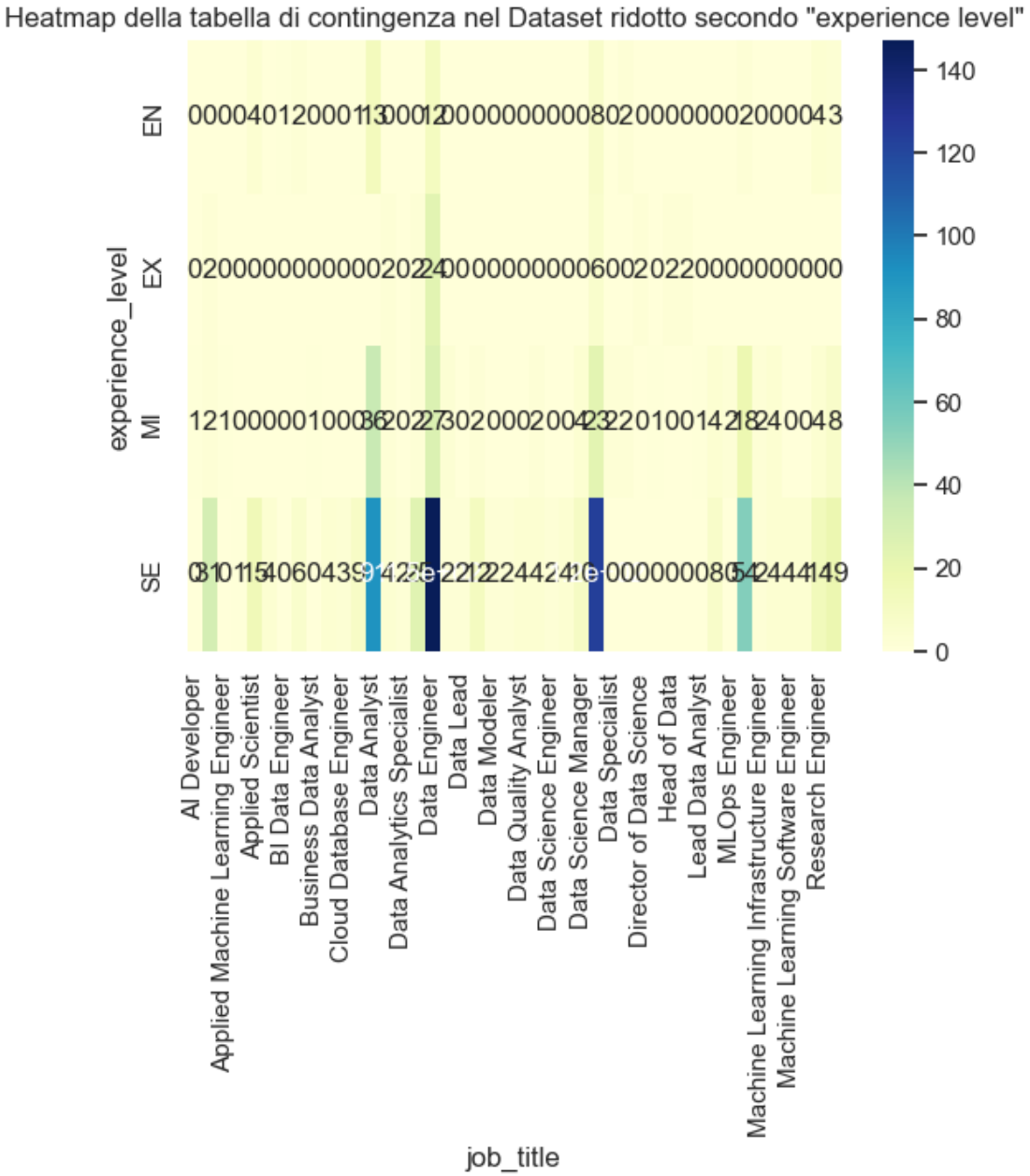
1.5.2 PER IL DATASET RIDOTTO

[ ]:
```
import pandas as pd
from scipy.stats import chi2_contingency

# Calcola la tabella di contingenza tra 'salary' e 'categoria'
contingency_table3 = pd.crosstab(dataset_ridotto['experience_level'],␣
 ↪dataset_ridotto['job_title'])

# Esegui il test del chi-quadro
chi2, p_value3, dof, expected = chi2_contingency(contingency_table3)

# Visualizza il p-value
print("P-value:", p_value3)
```

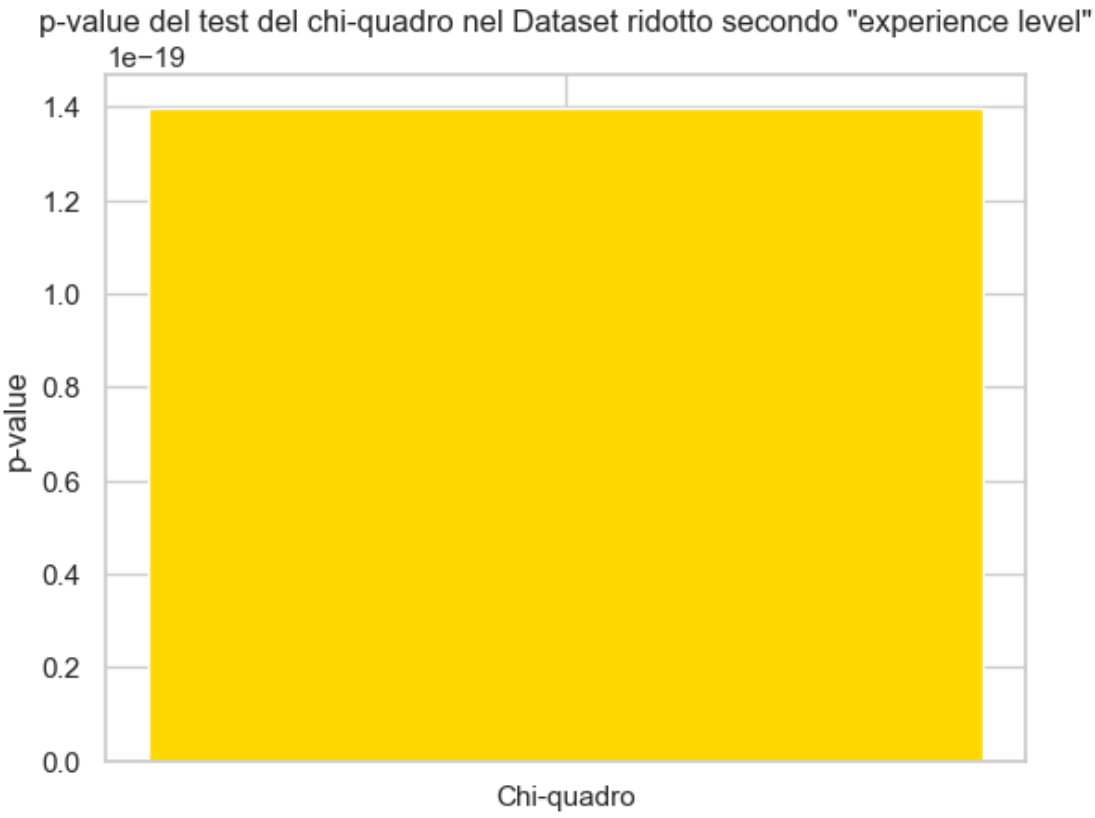
```
P-value: 1.3985820948525885e-19
```

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Utilizza seaborn per creare un heatmap della tabella di contingenza
sns.heatmap(contingency_table3, annot=True, cmap="YlGnBu")

# Aggiungi titolo
plt.title('Heatmap della tabella di contingenza nel Dataset ridotto secondo␣
 ↪\"experience level\"')

# Mostra il grafico
plt.show()
```



[ ]:
```
import matplotlib.pyplot as plt

# Crea un barplot del p-value
plt.bar(['Chi-quadro'], [p_value3], color='gold')

# Aggiungi titoli e label
plt.ylabel('p-value')
plt.title('p-value del test del chi-quadro nel Dataset ridotto secondo␣
 ↪\"experience level\"')

# Mostra il grafico
plt.show()
```

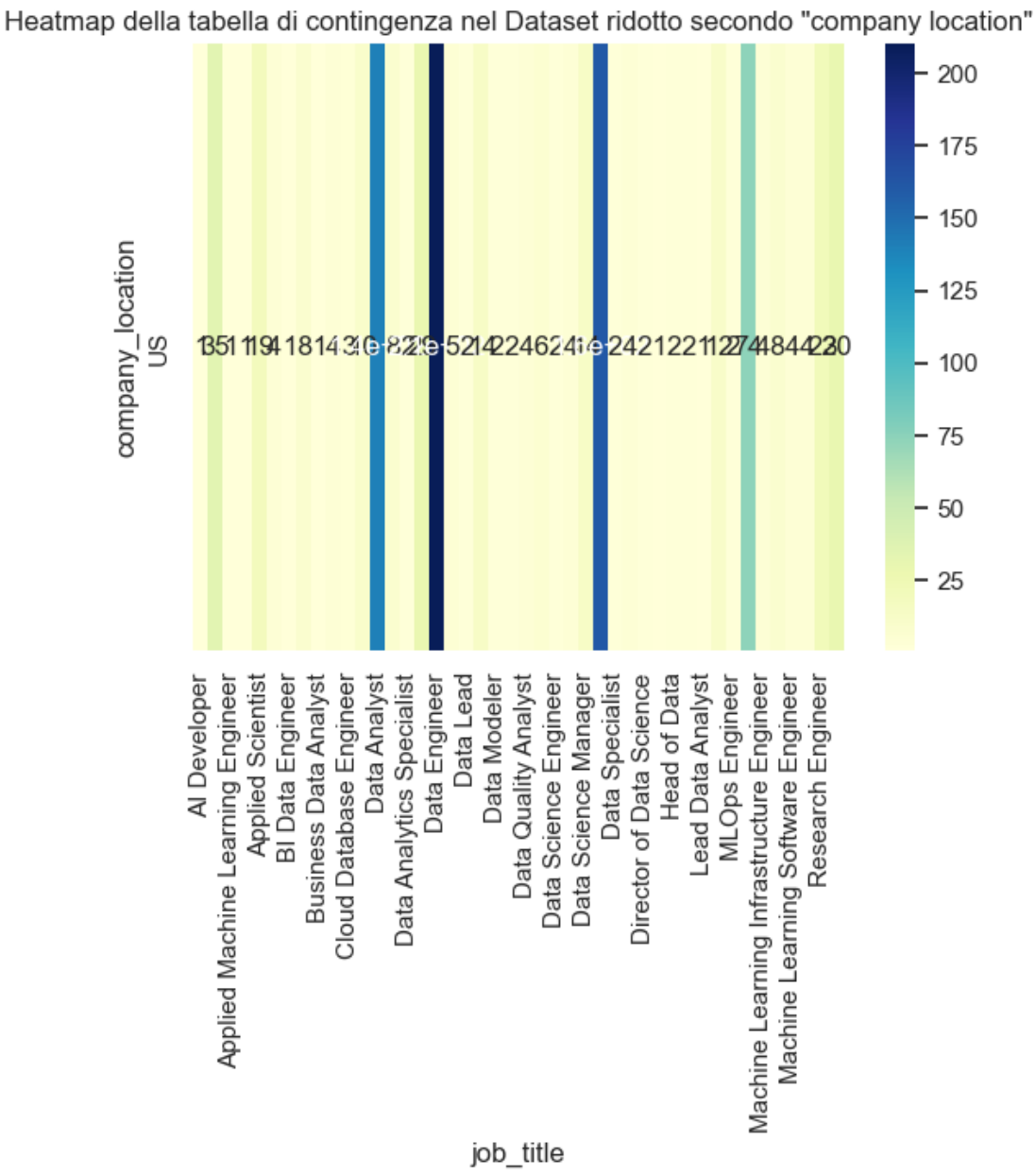


[ ]:
```
import pandas as pd
from scipy.stats import chi2_contingency

# Calcola la tabella di contingenza tra 'salary' e 'categoria'
contingency_table4 = pd.crosstab(dataset_ridotto['company_location'],␣
 ↪dataset_ridotto['job_title'])
```

```
# Esegui il test del chi-quadro
chi2, p_value4, dof, expected = chi2_contingency(contingency_table4)

# Visualizza il p-value
print("P-value:", p_value4)
```

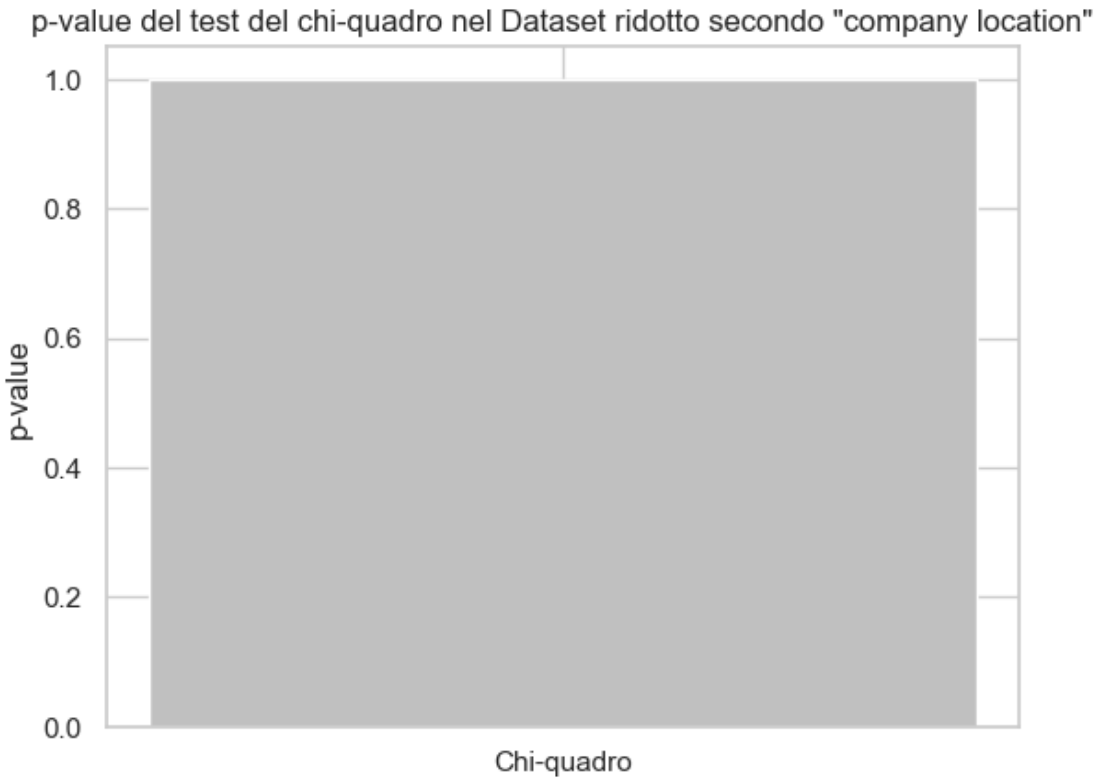
```
P-value: 1.0
```

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Utilizza seaborn per creare un heatmap della tabella di contingenza
sns.heatmap(contingency_table4, annot=True, cmap="YlGnBu")

# Aggiungi titolo
plt.title('Heatmap della tabella di contingenza nel Dataset ridotto secondo␣
 ↪\"company location\"')

# Mostra il grafico
plt.show()
```



[ ]:
```
import matplotlib.pyplot as plt

# Crea un barplot del p-value
plt.bar(['Chi-quadro'], [p_value4], color='silver')

# Aggiungi titoli e label
plt.ylabel('p-value')
plt.title('p-value del test del chi-quadro nel Dataset ridotto secondo␣
 ↪\"company location\"')
```

```
# Mostra il grafico
plt.show()
```



1.6 FASE 6: L'ANALISI DELLA PRESENZA DI NAN NEL DATASET, LA GESTIONE DI QUEST'ULTIMI ED EVENTUALI GRAFICI

[ ]:
```
# Calcolo del totale delle righe con dati mancanti
totale_dati_mancanti_dataset = dataset.isnull().any(axis=1).sum() # Calcola il␣
 ↪totale delle righe con almeno un dato mancante

# Determinazione delle colonne con dati mancanti
colonne_dati_mancanti_dataset = dataset.isnull().any(axis=0) # True se almeno␣
 ↪un valore nella colonna è mancante (None o NaN)
```

[ ]:
```
# Stampa delle colonne con dati mancanti e del totale dei dati mancanti
print("Colonne con i NaN nel Dataset originale:")
print(colonne_dati_mancanti_dataset)
print(f"Totale delle righe con i NaN nel Dataset originale:␣
 ↪{totale_dati_mancanti_dataset}")
```

```
Colonne con i NaN nel Dataset originale:
```

```
experience_level    False
job_title           False
salary              False
company_location    False
dtype: bool
Totale delle righe con i NaN nel Dataset originale: 0
```

[ ]:
```
# Calcolo del totale delle righe con dati mancanti
totale_dati_mancanti_dataset_ridotto = dataset_ridotto.isnull().any(axis=1).
 ↪sum() # Calcola il totale delle righe con almeno un dato mancante

# Determinazione delle colonne con dati mancanti
colonne_dati_mancanti_dataset_ridotto = dataset_ridotto.isnull().any(axis=0) #␣
 ↪True se almeno un valore nella colonna è mancante (None o NaN)
```

[ ]:
```
# Stampa delle colonne con dati mancanti e del totale dei dati mancanti
print("Colonne con i NaN nel Dataset ridotto:")
print(colonne_dati_mancanti_dataset_ridotto)
print(f"Totale delle righe con i NaN nel Dataset ridotto:␣
 ↪{totale_dati_mancanti_dataset_ridotto}")
```

```
Colonne con i NaN nel Dataset ridotto:
experience_level    False
job_title           False
salary              False
company_location    False
dtype: bool
Totale delle righe con i NaN nel Dataset ridotto: 0
```

1.7 FASE 7: L'ANALISI DELLA PRESENZA DI OUTLIERS NEL DATASET, LA GESTIONE DI QUEST'ULTIMI ED EVENTUALI GRAFICI

[ ]:
```
# la formula della deviazione standard è:   = √(Σ(xi - x̄)² / n)
    # √ = radice quadrata
    # Σ = sommatoria di tutti gli elementi dentro la parentesi quadra
    # xi = sono i singoli valori dei dati
    # x̄ = è la media dei dati
    # n = è il numero totale di dati
```

[ ]:
```
# Calcolare la media del Dataset
mean_value_dataset = dataset["salary"].mean()
print("La media dei valori del Dataset originario nella Feature \"salary\" è:")
print(mean_value_dataset)
```

```
La media dei valori del Dataset originario nella Feature "salary" è:
166798.63123236125
```

[ ]:
```
# Calcolare la media del Dataset ridotto
mean_value_dataset_ridotto = dataset_ridotto["salary"].mean()
print("La media dei valori del Dataset ridotto nella Feature \"salary\" è:")
print(mean_value_dataset_ridotto)
```

```
La media dei valori del Dataset ridotto nella Feature "salary" è:
160131.97462514418
```

[ ]:
```
# Calcolare la deviazione standard del Dataset
std_dev_dataset = dataset["salary"].std()
print("La deviazione standard del Dataset originario nella Feature \"salary\" è:
 ↪")
print(std_dev_dataset)
```

```
La deviazione standard del Dataset originario nella Feature "salary" è:
205073.26639455935
```

[ ]:
```
# Calcolare la deviazione standard del Dataset ridotto
std_dev_dataset_ridotto = dataset_ridotto["salary"].std()
print("La deviazione standard del Dataset originario nella Feature \"salary\" è:
 ↪")
print(std_dev_dataset_ridotto)
```

```
La deviazione standard del Dataset originario nella Feature "salary" è:
60578.67710373906
```

[ ]:
```
#Identifica gli outliers consiederando +3 sigma dalla media
outliers_dataset=dataset[(dataset["salary"]>mean_value_dataset+3*std_dev_dataset)␣
 ↪| (dataset["salary"]<mean_value_dataset-3*std_dev_dataset)] # Serve per␣
 ↪controllare la presenza effettiva di Outliers comparando se i valori della␣
 ↪Feature "salary" si discostano di 3 (sigma) dalla media
outliers_dataset
```

[ ]:

	experience_level	job_title	salary	company_location
156	MI	Applied Data Scientist	1700000	IN
217	EN	Data Engineer	1400000	IN
528	SE	AI Scientist	1500000	IL
735	MI	Data Scientist	1400000	IN
738	MI	Lead Data Analyst	1500000	IN
988	SE	Data Analyst	1300000	IN
998	SE	Data Science Consultant	1000000	TH
1230	EN	Data Scientist	800000	IN
1260	MI	Product Data Analyst	1350000	IN
1341	EN	Data Scientist	1050000	IN
1462	MI	Head of Data Science	5000000	IN
1512	EN	Data Scientist	1060000	IN
1549	MI	Data Analytics Lead	1440000	SG
1595	MI	Data Scientist	840000	TH

```
[ ]: outliers_dataset_ordinati= outliers_dataset.sort_values(by="salary",␣
      ↪ascending=False) # Se ascending è su False vuol dire che non è dal minore al␣
      ↪maggiore bensì il contrario, quindi che è dal maggiore al minore
     outliers_dataset_ordinati
```

```
[ ]:      experience_level                job_title   salary company_location
     1462               MI     Head of Data Science  5000000                IN
     156                MI   Applied Data Scientist  1700000                IN
     528                SE              AI Scientist  1500000                IL
     738                MI        Lead Data Analyst  1500000                IN
     1549               MI      Data Analytics Lead  1440000                SG
     217                EN            Data Engineer  1400000                IN
     735                MI           Data Scientist  1400000                IN
     1260               MI     Product Data Analyst  1350000                IN
     988                SE             Data Analyst  1300000                IN
     1512               EN           Data Scientist  1060000                IN
     1341               EN           Data Scientist  1050000                IN
     998                SE  Data Science Consultant  1000000                TH
     1595               MI           Data Scientist   840000                TH
     1230               EN           Data Scientist   800000                IN
```

```
[ ]: #Identifica gli outliers consiederando +3 sigma dalla media
     outliers_dataset_ridotto=dataset_ridotto[(dataset_ridotto["salary"]>mean_value_dataset_ridotto
      ↪|␣
      ↪(dataset_ridotto["salary"]<mean_value_dataset_ridotto-3*std_dev_dataset_ridotto)]
     outliers_dataset_ridotto
```

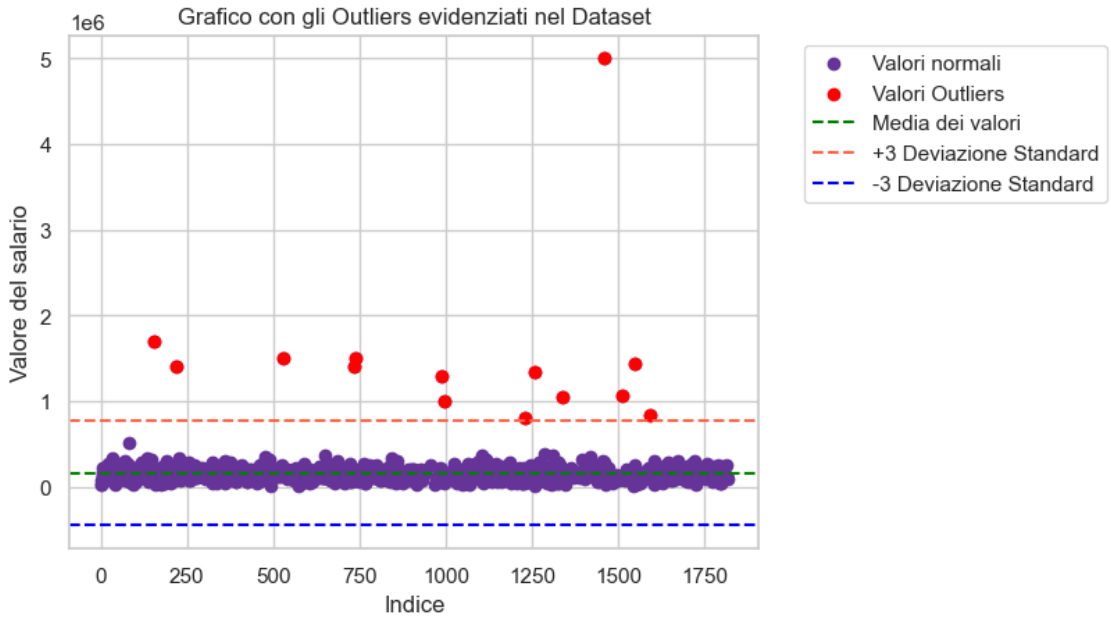
```
[ ]:      experience_level                 job_title  salary company_location
     33                 SE   Computer Vision Engineer  342810                US
     133                SE  Machine Learning Engineer  342300                US
     478                EX   Director of Data Science  353200                US
     649                SE             Data Architect  376080                US
     1105               SE             Data Scientist  370000                US
     1288               SE               Data Analyst  385000                US
     1311               SE         Research Scientist  370000                US
     1421               SE          Applied Scientist  350000                US
```

```
[ ]: outliers_dataset_ridotto_ordinati= outliers_dataset_ridotto.
      ↪sort_values(by="salary", ascending=False) # Se ascending è su False vuol␣
      ↪dire che non è dal minore al maggiore bensì il contrario, quindi che è dal␣
      ↪maggiore al minore
     outliers_dataset_ridotto_ordinati
```

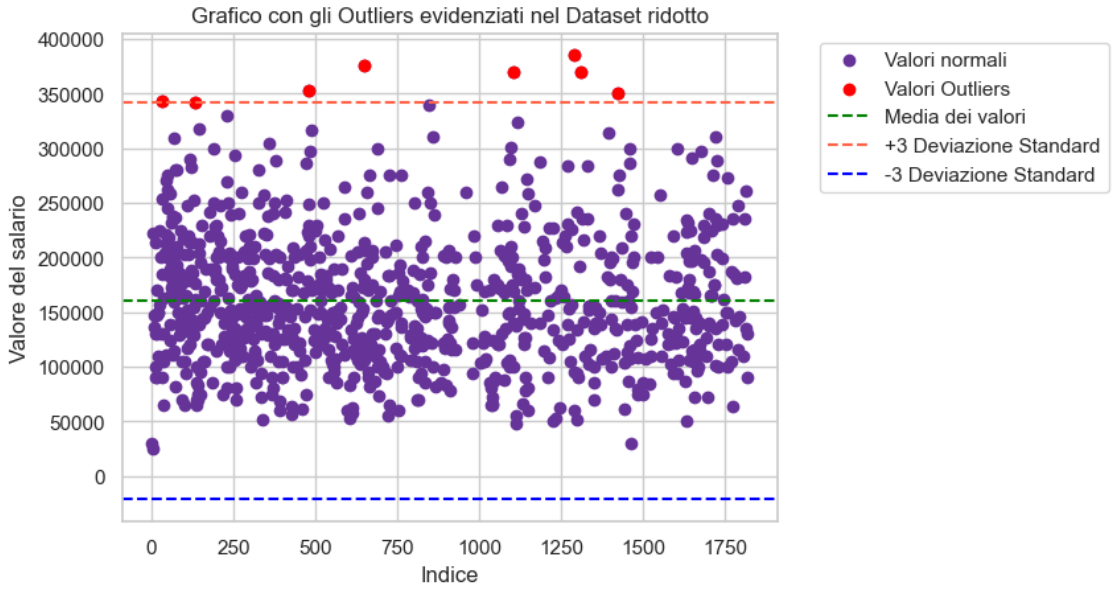
```
[ ]:      experience_level                 job_title  salary company_location
     1288               SE               Data Analyst  385000                US
     649                SE             Data Architect  376080                US
     1105               SE             Data Scientist  370000                US
```

```
1311              SE         Research Scientist  370000              US
478               EX   Director of Data Science  353200              US
1421              SE          Applied Scientist  350000              US
33                SE   Computer Vision Engineer  342810              US
133               SE  Machine Learning Engineer  342300              US
```

[ ]:
```python
# Crea un grafico a dispersione
plt.scatter(dataset.index, dataset['salary'], label='Valori normali',␣
 ↪color="rebeccapurple")

# Evidenzia gli outliers nel grafico con un colore diverso
plt.scatter(outliers_dataset.index, outliers_dataset['salary'], color='red',␣
 ↪label='Valori Outliers')

# Aggiungi la media e la deviazione standard al grafico
plt.axhline(y=mean_value_dataset, color='green', linestyle='--', label='Media␣
 ↪dei valori')
plt.axhline(y=mean_value_dataset + 3 * std_dev_dataset, color='tomato',␣
 ↪linestyle='--', label='+3 Deviazione Standard')
plt.axhline(y=mean_value_dataset - 3 * std_dev_dataset, color='blue',␣
 ↪linestyle='--', label='-3 Deviazione Standard')

# Aggiungi etichette e legenda al grafico
plt.xlabel('Indice')
plt.ylabel('Valore del salario')
plt.title('Grafico con gli Outliers evidenziati nel Dataset')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Mostra il grafico
plt.show()
```

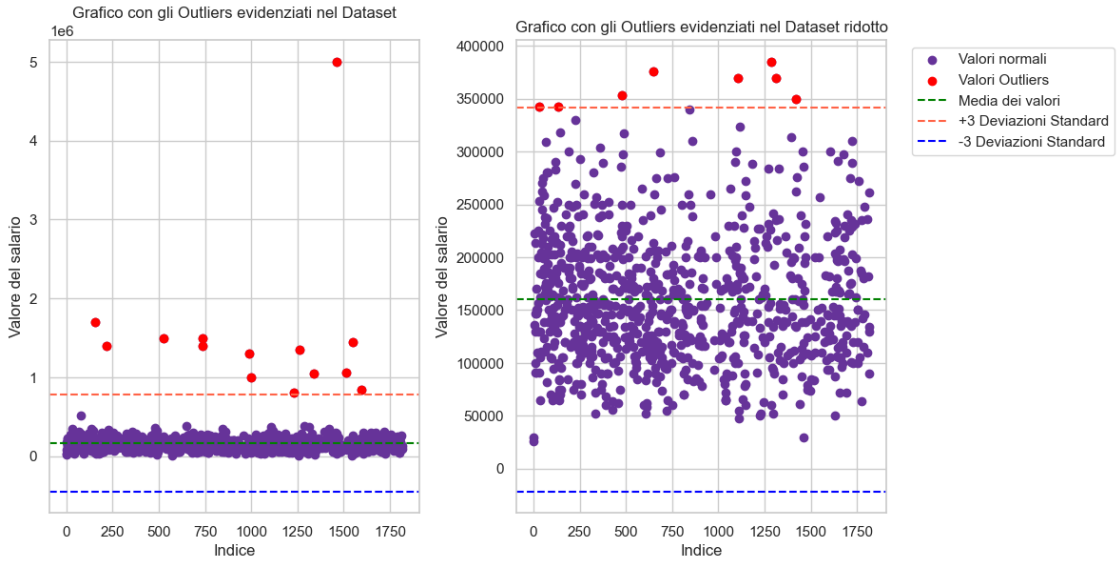


[ ]:
```
# Crea un grafico a dispersione
plt.scatter(dataset_ridotto.index, dataset_ridotto['salary'], label='Valori␣
 ↪normali', color="rebeccapurple")

# Evidenzia gli outliers nel grafico con un colore diverso
plt.scatter(outliers_dataset_ridotto.index, outliers_dataset_ridotto['salary'],␣
 ↪color='red', label='Valori Outliers')

# Aggiungi la media e la deviazione standard al grafico
plt.axhline(y=mean_value_dataset_ridotto, color='green', linestyle='--',␣
 ↪label='Media dei valori')
plt.axhline(y=mean_value_dataset_ridotto + 3 * std_dev_dataset_ridotto,␣
 ↪color='tomato', linestyle='--', label='+3 Deviazione Standard')
plt.axhline(y=mean_value_dataset_ridotto - 3 * std_dev_dataset_ridotto,␣
 ↪color='blue', linestyle='--', label='-3 Deviazione Standard')

# Aggiungi etichette e legenda al grafico
plt.xlabel('Indice')
plt.ylabel('Valore del salario')
plt.title('Grafico con gli Outliers evidenziati nel Dataset ridotto')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Mostra il grafico
plt.show()
```



[ ]:
```
# Importa la libreria matplotlib
import matplotlib.pyplot as plt

# Crea una figura e due assi (subplot)
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

# Grafico con outliers nel dataset originale
axs[0].scatter(dataset.index, dataset['salary'], label='Valori normali',␣
 ↪color="rebeccapurple")
axs[0].scatter(outliers_dataset.index, outliers_dataset['salary'], color='red',␣
 ↪label='Valori Outliers')
axs[0].axhline(y=mean_value_dataset, color='green', linestyle='--',␣
 ↪label='Media dei valori')
axs[0].axhline(y=mean_value_dataset + 3 * std_dev_dataset, color='tomato',␣
 ↪linestyle='--', label='+3 Deviazione Standard')
axs[0].axhline(y=mean_value_dataset - 3 * std_dev_dataset, color='blue',␣
 ↪linestyle='--', label='-3 Deviazione Standard')
axs[0].set_xlabel('Indice')
axs[0].set_ylabel('Valore del salario')
axs[0].set_title('Grafico con gli Outliers evidenziati nel Dataset')

# Grafico con outliers nel dataset ridotto
axs[1].scatter(dataset_ridotto.index, dataset_ridotto['salary'], label='Valori␣
 ↪normali', color="rebeccapurple")
axs[1].scatter(outliers_dataset_ridotto.index,␣
 ↪outliers_dataset_ridotto['salary'], color='red', label='Valori Outliers')
```

```
axs[1].axhline(y=mean_value_dataset_ridotto, color='green', linestyle='--',␣
 ↪label='Media dei valori')
axs[1].axhline(y=mean_value_dataset_ridotto + 3 * std_dev_dataset_ridotto,␣
 ↪color='tomato', linestyle='--', label='+3 Deviazioni Standard')
axs[1].axhline(y=mean_value_dataset_ridotto - 3 * std_dev_dataset_ridotto,␣
 ↪color='blue', linestyle='--', label='-3 Deviazioni Standard')
axs[1].set_xlabel('Indice')
axs[1].set_ylabel('Valore del salario')
axs[1].set_title('Grafico con gli Outliers evidenziati nel Dataset ridotto')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Regola la disposizione e lo spazio tra i subplot
plt.tight_layout()

# Mostra i grafici
plt.show()
```



[ ]:
```
# Definisci il numero minimo di features che devono superare la soglia per␣
 ↪considerare un dato un outlier
min_features_threshold = 1
k = 3  # intervallo di confidenza

# Lista per salvare gli indici degli outliers
outlier_indices_dataset = []

# Calcola la media e la deviazione standard della feature "salary"
mean_salary_dataset = dataset['salary'].mean()
std_dev_salary_dataset = dataset['salary'].std()
```

```
# Identifica gli outliers per la feature "salary"
dataset['Outlier_salary'] = (dataset['salary'] > mean_salary_dataset + k *␣
 ↪std_dev_salary_dataset) | (dataset['salary'] < mean_salary_dataset - k *␣
 ↪std_dev_salary_dataset)

dataset
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1678530265.py:13:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset['Outlier_salary'] = (dataset['salary'] > mean_salary_dataset + k *
std_dev_salary_dataset) | (dataset['salary'] < mean_salary_dataset - k *
std_dev_salary_dataset)
```

[ ]:

	experience_level	job_title	salary	company_location	\
0	SE	Principal Data Scientist	80000	ES	
1	MI	ML Engineer	30000	US	
2	MI	ML Engineer	25500	US	
3	SE	Data Scientist	175000	CA	
4	SE	Data Scientist	120000	CA	
…	…	…	…	…	
1809	SE	Data Engineer	182000	US	
1814	SE	Machine Learning Engineer	261500	US	
1815	SE	Machine Learning Engineer	134500	US	
1817	MI	Data Scientist	130000	US	
1818	MI	Data Scientist	90000	US	

	Outlier_salary
0	False
1	False
2	False
3	False
4	False
…	…
1809	False
1814	False
1815	False
1817	False
1818	False

[1063 rows x 5 columns]

[ ]:
```
# Definisci il numero minimo di features che devono superare la soglia per␣
 ↪considerare un dato un outlier
min_features_threshold = 1
k = 3  # intervallo di confidenza

# Lista per salvare gli indici degli outliers
outlier_indices_dataset_ridotto = []

# Calcola la media e la deviazione standard della feature "salary"
mean_salary_dataset_ridotto = dataset_ridotto['salary'].mean()
std_dev_salary_dataset_ridotto = dataset_ridotto['salary'].std()

# Identifica gli outliers per la feature "salary"
dataset_ridotto['Outlier_salary'] = (dataset_ridotto['salary'] >␣
 ↪mean_salary_dataset_ridotto + k * std_dev_salary_dataset_ridotto) |␣
 ↪(dataset_ridotto['salary'] < mean_salary_dataset_ridotto - k *␣
 ↪std_dev_salary_dataset_ridotto)

dataset_ridotto
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\2167628478.py:13:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset_ridotto['Outlier_salary'] = (dataset_ridotto['salary'] >
mean_salary_dataset_ridotto + k * std_dev_salary_dataset_ridotto) |
(dataset_ridotto['salary'] < mean_salary_dataset_ridotto - k *
std_dev_salary_dataset_ridotto)
```

[ ]:

	experience_level	job_title	salary	company_location	\
1	MI	ML Engineer	30000	US	
2	MI	ML Engineer	25500	US	
5	SE	Applied Scientist	222200	US	
6	SE	Applied Scientist	136000	US	
9	SE	Data Scientist	147100	US	
…	…	…	…	…	
1809	SE	Data Engineer	182000	US	
1814	SE	Machine Learning Engineer	261500	US	
1815	SE	Machine Learning Engineer	134500	US	
1817	MI	Data Scientist	130000	US	
1818	MI	Data Scientist	90000	US	

	Outlier_salary
1	False

```
2              False
5              False
6              False
9              False
…              …
1809           False
1814           False
1815           False
1817           False
1818           False

[867 rows x 5 columns]
```

[ ]:
```
#Elimina le righe corrispondenti agli outliers quelli che hanno una features␣
 ↪fuoriscala
outliers_dataset = dataset['Num_Outliers_nella_riga'] = dataset.
 ↪filter(like='Outlier_').sum(axis=1) # Serve per contare quanti Outliers ci␣
 ↪sono per ogni riga ed essendoci una sola Feature numerica il valore␣
 ↪obbligatoriamente sarà pari a 0 o a 1, solo in questo caso
dataset
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1611520539.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  outliers_dataset = dataset['Num_Outliers_nella_riga'] =
dataset.filter(like='Outlier_').sum(axis=1) # Serve per contare quanti Outliers
ci sono per ogni riga ed essendoci una sola Feature numerica il valore
obbligatoriamente sarà pari a 0 o a 1, solo in questo caso
```

[ ]:
```
     experience_level                job_title  salary company_location  \
0                  SE   Principal Data Scientist   80000               ES
1                  MI               ML Engineer   30000               US
2                  MI               ML Engineer   25500               US
3                  SE             Data Scientist  175000               CA
4                  SE             Data Scientist  120000               CA
…                 …                          …     …                 …
1809               SE              Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI             Data Scientist  130000               US
1818               MI             Data Scientist   90000               US

      Outlier_salary  Num_Outliers_nella_riga
```

```
0              False                        0
1              False                        0
2              False                        0
3              False                        0
4              False                        0
…              …                          …
1809           False                        0
1814           False                        0
1815           False                        0
1817           False                        0
1818           False                        0

[1063 rows x 6 columns]
```

[ ]:
```
#Elimina le righe corrispondenti agli outliers quelli che hanno una features␣
 ↪fuoriscala
outliers_dataset_ridotto = dataset_ridotto['Num_Outliers_nella_riga'] =␣
 ↪dataset_ridotto.filter(like='Outlier_').sum(axis=1)
dataset_ridotto
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1806719169.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  outliers_dataset_ridotto = dataset_ridotto['Num_Outliers_nella_riga'] =
dataset_ridotto.filter(like='Outlier_').sum(axis=1)
```

[ ]:
```
     experience_level                  job_title  salary company_location  \
1                  MI               ML Engineer   30000               US
2                  MI               ML Engineer   25500               US
5                  SE          Applied Scientist  222200               US
6                  SE          Applied Scientist  136000               US
9                  SE             Data Scientist  147100               US
…                 …                          …     …                  …
1809               SE              Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI             Data Scientist  130000               US
1818               MI             Data Scientist   90000               US

      Outlier_salary  Num_Outliers_nella_riga
1              False                        0
2              False                        0
5              False                        0
```

```
6              False                        0
9              False                        0
…              …                          …
1809           False                        0
1814           False                        0
1815           False                        0
1817           False                        0
1818           False                        0

[867 rows x 6 columns]
```

[ ]:
```
# Filtra i dati per mantenere solo le righe con almeno il numero minimo di␣
 ↪features superanti la soglia
outliers_dataset = dataset[dataset['Num_Outliers_nella_riga'] >=␣
 ↪min_features_threshold]
outliers_dataset
```

[ ]:
```
     experience_level                job_title   salary company_location  \
156                MI   Applied Data Scientist  1700000               IN
217                EN            Data Engineer  1400000               IN
528                SE              AI Scientist  1500000               IL
735                MI           Data Scientist  1400000               IN
738                MI        Lead Data Analyst  1500000               IN
988                SE             Data Analyst  1300000               IN
998                SE  Data Science Consultant  1000000               TH
1230               EN           Data Scientist   800000               IN
1260               MI     Product Data Analyst  1350000               IN
1341               EN           Data Scientist  1050000               IN
1462               MI     Head of Data Science  5000000               IN
1512               EN           Data Scientist  1060000               IN
1549               MI      Data Analytics Lead  1440000               SG
1595               MI           Data Scientist   840000               TH

     Outlier_salary  Num_Outliers_nella_riga
156            True                        1
217            True                        1
528            True                        1
735            True                        1
738            True                        1
988            True                        1
998            True                        1
1230           True                        1
1260           True                        1
1341           True                        1
1462           True                        1
1512           True                        1
1549           True                        1
```

```
1595            True                          1
```

[ ]:
```
# Filtra i dati per mantenere solo le righe con almeno il numero minimo di␣
 ↪features superanti la soglia
outliers_dataset_ridotto =␣
 ↪dataset_ridotto[dataset_ridotto['Num_Outliers_nella_riga'] >=␣
 ↪min_features_threshold]
outliers_dataset_ridotto
```

[ ]:
```
     experience_level                 job_title  salary company_location  \
33                 SE   Computer Vision Engineer  342810               US
133                SE  Machine Learning Engineer  342300               US
478                EX   Director of Data Science  353200               US
649                SE             Data Architect  376080               US
1105               SE             Data Scientist  370000               US
1288               SE               Data Analyst  385000               US
1311               SE         Research Scientist  370000               US
1421               SE          Applied Scientist  350000               US

      Outlier_salary  Num_Outliers_nella_riga
33              True                        1
133             True                        1
478             True                        1
649             True                        1
1105            True                        1
1288            True                        1
1311            True                        1
1421            True                        1
```

[ ]:
```
dataset_filtered = dataset[dataset['Outlier_salary'] == False]
dataset_filtered
```

[ ]:
```
     experience_level                 job_title  salary company_location  \
0                  SE   Principal Data Scientist   80000               ES
1                  MI               ML Engineer   30000               US
2                  MI               ML Engineer   25500               US
3                  SE             Data Scientist  175000               CA
4                  SE             Data Scientist  120000               CA
…                  …                         …       …                …
1809               SE              Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI             Data Scientist  130000               US
1818               MI             Data Scientist   90000               US

      Outlier_salary  Num_Outliers_nella_riga
0              False                        0
```

```
1              False                        0
2              False                        0
3              False                        0
4              False                        0
…              …                          …
1809           False                        0
1814           False                        0
1815           False                        0
1817           False                        0
1818           False                        0

[1049 rows x 6 columns]
```

[ ]:
```
dataset_ridotto_filtered = dataset_ridotto[dataset_ridotto['Outlier_salary'] ==␣
 ↪False]
dataset_ridotto_filtered
```

[ ]:
```
     experience_level                job_title  salary company_location  \
1                  MI               ML Engineer   30000               US
2                  MI               ML Engineer   25500               US
5                  SE         Applied Scientist  222200               US
6                  SE         Applied Scientist  136000               US
9                  SE            Data Scientist  147100               US
…                 …                         …     …                 …
1809               SE             Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI            Data Scientist  130000               US
1818               MI            Data Scientist   90000               US

      Outlier_salary  Num_Outliers_nella_riga
1              False                        0
2              False                        0
5              False                        0
6              False                        0
9              False                        0
…              …                          …
1809           False                        0
1814           False                        0
1815           False                        0
1817           False                        0
1818           False                        0

[859 rows x 6 columns]
```

[ ]:
```
# Filtra i dati per mantenere solo le righe con almeno il numero minimo di␣
 ↪features superanti la soglia
```

```
outliers_dataset_filtered =␣
 ↪dataset_filtered[dataset_filtered['Num_Outliers_nella_riga'] >=␣
 ↪min_features_threshold]
outliers_dataset_filtered
```

[ ]:
```
Empty DataFrame
Columns: [experience_level, job_title, salary, company_location, Outlier_salary,
Num_Outliers_nella_riga]
Index: []
```

[ ]:
```
# Filtra i dati per mantenere solo le righe con almeno il numero minimo di␣
 ↪features superanti la soglia
outliers_dataset_ridotto_filtered =␣
 ↪dataset_ridotto_filtered[dataset_ridotto_filtered['Num_Outliers_nella_riga']␣
 ↪>= min_features_threshold]
outliers_dataset_ridotto_filtered
```

[ ]:
```
Empty DataFrame
Columns: [experience_level, job_title, salary, company_location, Outlier_salary,
Num_Outliers_nella_riga]
Index: []
```

[ ]:
```
print(dataset.shape)
print(dataset_filtered.shape)
print(dataset_ridotto.shape)
print(dataset_ridotto_filtered.shape)
```

```
(1063, 6)
(1049, 6)
(867, 6)
(859, 6)
```

[ ]:
```
dataset=dataset_filtered
dataset_ridotto=dataset_ridotto_filtered
```

[ ]:
```
# Rimuovi colonne ausiliarie
dataset.drop(dataset.filter(like='Outlier_').columns, axis=1, inplace=True) #␣
 ↪Questo serve per filtrare e succesivamente elimanare tutte quelle Feature␣
 ↪che iniziano con quel determinato suffiso, che nel caso del Dataset in␣
 ↪questione è "salary" che è l'unica Feature numerica
dataset.drop('Num_Outliers_nella_riga', axis=1, inplace=True) # Drop vuol dire␣
 ↪buttare, quindi elimina in questo caso una Feature mentre axis pari a 1␣
 ↪indica una Feature e infine "inplace" indica che il Dataset viene␣
 ↪sovrascritto con le nuove modifiche
dataset
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1542602447.py:2:
SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset.drop(dataset.filter(like='Outlier_').columns, axis=1, inplace=True) #
Questo serve per filtrare e succesivamente elimanare tutte quelle Feature che
iniziano con quel determinato suffiso, che nel caso del Dataset in questione è
"salary" che è l'unica Feature numerica
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1542602447.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset.drop('Num_Outliers_nella_riga', axis=1, inplace=True) # Drop vuol dire
buttare, quindi elimina in questo caso una Feature mentre axis pari a 1 indica
una Feature e infine "inplace" indica che il Dataset viene sovrascritto con le
nuove modifiche
```

[ ]:

	experience_level	job_title	salary	company_location
0	SE	Principal Data Scientist	80000	ES
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
3	SE	Data Scientist	175000	CA
4	SE	Data Scientist	120000	CA
…	…	…	…	…
1809	SE	Data Engineer	182000	US
1814	SE	Machine Learning Engineer	261500	US
1815	SE	Machine Learning Engineer	134500	US
1817	MI	Data Scientist	130000	US
1818	MI	Data Scientist	90000	US

[1049 rows x 4 columns]

[ ]:
```
# Rimuovi colonne ausiliarie
dataset_ridotto.drop(dataset_ridotto.filter(like='Outlier_').columns, axis=1,␣
  ↪inplace=True)
dataset_ridotto.drop('Num_Outliers_nella_riga', axis=1, inplace=True)
dataset_ridotto
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\3394655298.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset_ridotto.drop(dataset_ridotto.filter(like='Outlier_').columns, axis=1,
inplace=True)
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\3394655298.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset_ridotto.drop('Num_Outliers_nella_riga', axis=1, inplace=True)
```

[ ]:

	experience_level	job_title	salary	company_location
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
5	SE	Applied Scientist	222200	US
6	SE	Applied Scientist	136000	US
9	SE	Data Scientist	147100	US
…	…	…	…	…
1809	SE	Data Engineer	182000	US
1814	SE	Machine Learning Engineer	261500	US
1815	SE	Machine Learning Engineer	134500	US
1817	MI	Data Scientist	130000	US
1818	MI	Data Scientist	90000	US

[859 rows x 4 columns]

[ ]:
```
dataset
```

[ ]:

	experience_level	job_title	salary	company_location
0	SE	Principal Data Scientist	80000	ES
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
3	SE	Data Scientist	175000	CA
4	SE	Data Scientist	120000	CA
…	…	…	…	…
1809	SE	Data Engineer	182000	US
1814	SE	Machine Learning Engineer	261500	US
1815	SE	Machine Learning Engineer	134500	US
1817	MI	Data Scientist	130000	US
1818	MI	Data Scientist	90000	US

[1049 rows x 4 columns]

[ ]:
```
dataset_ridotto
```

[ ]:

	experience_level	job_title	salary	company_location
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
5	SE	Applied Scientist	222200	US
6	SE	Applied Scientist	136000	US
9	SE	Data Scientist	147100	US

```
…                …                              …      …                …
1809               SE              Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI             Data Scientist  130000               US
1818               MI             Data Scientist   90000               US

[859 rows x 4 columns]
```

1.8 FASE 8: LO SCALING ED ENCODING DEI DATI NELLE FEATURE (CON I GRAFICI)

1.8.1 LE OPERAZIONI PRELIMINARI

[ ]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler

# Escludi le colonne non numeriche dal DataFrame
Feature_numeriche_dataset = dataset.select_dtypes(include=['number']).columns
dataset_Feature_numeriche = dataset[Feature_numeriche_dataset]

dataset_Feature_numeriche
```

[ ]:
```
      salary
0      80000
1      30000
2      25500
3     175000
4     120000
…        …
1809  182000
1814  261500
1815  134500
1817  130000
1818   90000

[1049 rows x 1 columns]
```

[ ]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler

# Escludi le colonne non numeriche dal DataFrame
Feature_numeriche_dataset_ridotto = dataset_ridotto.
 ↪select_dtypes(include=['number']).columns
dataset_ridotto_Feature_numeriche =␣
 ↪dataset_ridotto[Feature_numeriche_dataset_ridotto]

dataset_ridotto_Feature_numeriche
```

[ ]:
```
      salary
1      30000
2      25500
5     222200
6     136000
9     147100
…        …
1809  182000
1814  261500
1815  134500
1817  130000
1818   90000

[859 rows x 1 columns]
```

1.8.2 IL MIN-MAX SCALING

Il Min-Max scaling acquisisce il valore Max (il più alto valore della Feature "salary" in questo caso) e gli cambia il valore a 1. Dopo acquisisce il valore Min, quindi quello minimo sempre della Feature "salary", e lo transforma in 0. Infine gli altri valori vengono scalati tra 0 e 1 (esclusi)

La sua formula vera e propria è:

x_scalata = (x - valore_minimo_di_x)/(valore_massimo_di_x - valore_minimo_di_x)

[ ]:
```
# Min-Max scaling solo delle colonne numeriche
min_max_scaling_dataset = MinMaxScaler()
min_max_scaling_dati_dataset = min_max_scaling_dataset.
 ↪fit_transform(dataset_Feature_numeriche)
min_max_scaling_dataset_numerico = pd.DataFrame(min_max_scaling_dati_dataset,␣
 ↪columns=dataset_Feature_numeriche.columns)

# Per provare l'effettiva riuscita dello Min Max Scaling bisogna ricavare i␣
 ↪primi numeri maggiori e minori del nuovo Dataset
min_row_dataset = min_max_scaling_dataset_numerico.
 ↪iloc[min_max_scaling_dataset_numerico.min(axis=1).idxmin()] # Utilizzare il␣
 ↪metodo iloc per indicare una riga o una Feature del DataFrame, in questo␣
 ↪caso non si può indicare direttamente il numero ma attraverso il comando min␣
 ↪si riesce a ricavare il numero minore del Dataset mentre con idxmin si␣
 ↪indica che dev'essere il primo di nunmero minore nel Dataset
max_row_dataset = min_max_scaling_dataset_numerico.
 ↪iloc[min_max_scaling_dataset_numerico.max(axis=1).idxmax()]
min_max_scaling_dataset_numerico
```

[ ]:
```
     salary
0  0.145129
1  0.045726
2  0.036779
```

```
3     0.333996
4     0.224652
…        …
1044  0.347913
1045  0.505964
1046  0.253479
1047  0.244533
1048  0.165010

[1049 rows x 1 columns]
```

[ ]:
```python
import matplotlib.pyplot as plt

colori=["red"]

# Creazione dei subplot per gli istogrammi
fig, axes = plt.subplots(nrows=1, ncols=len(min_max_scaling_dataset_numerico.
 ↪columns), figsize=(15, 5))

# Se c'è solo una colonna, axes non sarà una lista, quindi lo mettiamo in una␣
 ↪lista per iterare comunque
if len(min_max_scaling_dataset_numerico.columns) == 1:
    axes = [axes]

# Loop attraverso le colonne per disegnare gli istogrammi
for i, col in enumerate(min_max_scaling_dataset_numerico.columns):
    axes[i].hist(min_max_scaling_dataset_numerico[col], bins=20, alpha=0.7,␣
 ↪color=colori)
    axes[i].set_title('Distribuzione della Feature Salary')   # Imposta il␣
 ↪titolo
    axes[i].set_xlabel('Valori scalati')           # Imposta l'etichetta␣
 ↪sull'asse x
    axes[i].set_ylabel('Numero di persone')        # Imposta l'etichetta␣
 ↪sull'asse y

plt.tight_layout()
plt.show()
```

[ ]:
```
# Min-Max scaling solo delle colonne numeriche
min_max_scaling_dataset_ridotto = MinMaxScaler()
min_max_scaling_dati_dataset_ridotto = min_max_scaling_dataset_ridotto.
 ↪fit_transform(dataset_ridotto_Feature_numeriche)
min_max_scaling_dataset_ridotto_numerico = pd.
 ↪DataFrame(min_max_scaling_dati_dataset_ridotto,␣
 ↪columns=dataset_Feature_numeriche.columns)

# Per provare l'effettiva riuscita dello Min Max Scaling bisogna ricavare i␣
 ↪primi numeri maggiori e minori del nuovo Dataset
min_row_dataset_ridotto = min_max_scaling_dataset_ridotto_numerico.
 ↪iloc[min_max_scaling_dataset_ridotto_numerico.min(axis=1).idxmin()] #␣
 ↪Utilizzare il metodo iloc per indicare una riga o una Feature del DataFrame,␣
 ↪in questo caso non si può indicare direttamente il numero ma attraverso il␣
 ↪comando min si riesce a ricavare il numero minore del Dataset mentre con␣
 ↪idxmin si indica che dev'essere il primo di nunmero minore nel Dataset
max_row_dataset_ridotto = min_max_scaling_dataset_ridotto_numerico.
 ↪iloc[min_max_scaling_dataset_ridotto_numerico.max(axis=1).idxmax()]
min_max_scaling_dataset_ridotto_numerico
```

[ ]:

	salary
0	0.014308
1	0.000000
2	0.625437
3	0.351351
4	0.386645
..	…
854	0.497615
855	0.750397
856	0.346582
857	0.332273
858	0.205087

```
[859 rows x 1 columns]
```

[ ]:
```
print("Il valore minimo è:")
print(min_row_dataset_ridotto) # Il valore minimo
print("Il valore massimo è:")
print(max_row_dataset_ridotto) # Il valore massimo
```

```
Il valore minimo è:
salary    0.0
Name: 1, dtype: float64
Il valore massimo è:
salary    1.0
Name: 512, dtype: float64
```

[ ]:
```
print("Il valore minimo è:")
print(min_row_dataset) # Il valore minimo
print("Il valore massimo è:")
print(max_row_dataset) # Il valore massimo
```

```
Il valore minimo è:
salary    0.0
Name: 429, dtype: float64
Il valore massimo è:
salary    1.0
Name: 79, dtype: float64
```

[ ]:
```
print("Informazioni sulla riga del valore minimo:")
print(min_max_scaling_dataset_ridotto_numerico.iloc[1]) # Utilizzare il metodo␣
 ↪iloc per indicare il numero o il nome di una riga o di una Feature del␣
 ↪DataFrame
print("Informazioni sulla riga del valore massimo:")
print(min_max_scaling_dataset_ridotto_numerico.iloc[512])
```

```
Informazioni sulla riga del valore minimo:
salary    0.0
Name: 1, dtype: float64
Informazioni sulla riga del valore massimo:
salary    1.0
Name: 512, dtype: float64
```

[ ]:
```
print("Informazioni sulla riga del valore minimo:")
print(min_max_scaling_dataset_numerico.iloc[563]) # Utilizzare il metodo iloc␣
 ↪per indicare il numero o il nome di una riga o di una Feature del DataFrame
print("Informazioni sulla riga del valore massimo:")
print(min_max_scaling_dataset_numerico.iloc[79])
```

```
Informazioni sulla riga del valore minimo:
salary    0.236581
Name: 563, dtype: float64
```

```
Informazioni sulla riga del valore massimo:
salary    1.0
Name: 79, dtype: float64
```

1.8.3 LO Z-SCORE SCALING O LO STANDARD SCALING

Lo Z-score scaling o Standard scaling scala i valori usando la media dei valori e la deviazione standard applicando la seguente formula:

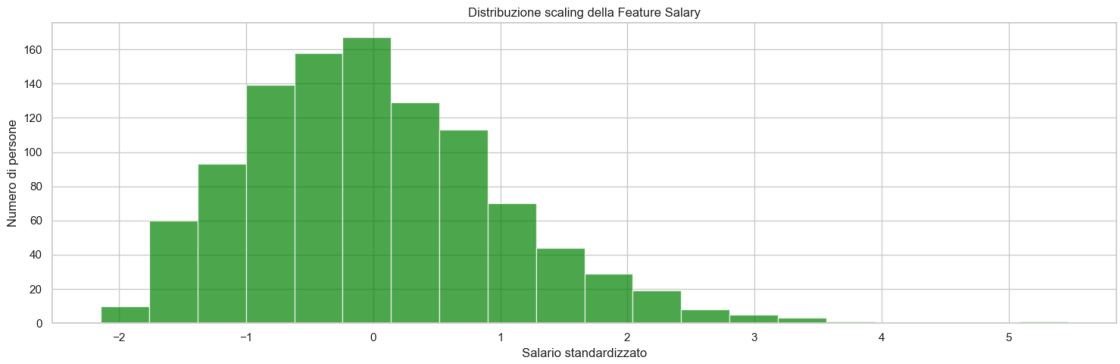
x_scalata = (x - valore_medio_di_x)/deviazione_standard_di_x

[ ]:
```
# Z-score scaling
standard_scaling_dataset = StandardScaler()
standard_scaling_dataset_dati = standard_scaling_dataset.
 ↪fit_transform(dataset_Feature_numeriche)
standard_scaling_dataset_numerico = pd.DataFrame(standard_scaling_dataset_dati,␣
 ↪columns=dataset_Feature_numeriche.columns)

standard_scaling_dataset_numerico
```

[ ]:
```
        salary
0    -1.039452
1    -1.796170
2    -1.864275
3     0.398313
4    -0.434077
…          …
1044  0.504254
1045  1.707436
1046 -0.214629
1047 -0.282733
1048 -0.888108

[1049 rows x 1 columns]
```

[ ]:
```
import matplotlib.pyplot as plt

colori=["green"]

# Numero di colonne nel DataFrame
num_cols_dataset = len(standard_scaling_dataset_numerico.columns)

# Creazione dei subplot per gli istogrammi
fig, axes = plt.subplots(nrows=1, ncols=num_cols_dataset, figsize=(15, 5))

# Se c'è solo una colonna, axes non sarà una lista, quindi lo mettiamo in una␣
 ↪lista per iterare comunque
if num_cols_dataset == 1:
```

```
    axes = [axes]

# Loop attraverso le colonne per disegnare gli istogrammi
for i, col in enumerate(standard_scaling_dataset_numerico.columns):
    axes[i].hist(standard_scaling_dataset_numerico[col], bins=20, alpha=0.7,␣
 ↪color=colori)
    axes[i].set_title(col)
    axes[i].set_title('Distribuzione scaling della Feature Salary')   # Imposta␣
 ↪il titolo
    axes[i].set_xlabel('Salario standardizzato')    # Imposta l'etichetta␣
 ↪sull'asse x
    axes[i].set_ylabel('Numero di persone')         # Imposta l'etichetta␣
 ↪sull'asse y

plt.tight_layout()
plt.show()
```



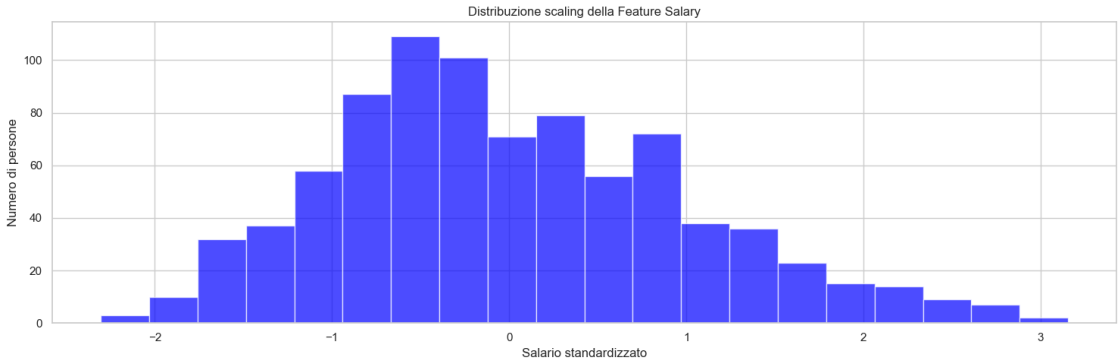
[ ]:
```
# Z-score scaling
standard_scaling_dataset_ridotto = StandardScaler()
standard_scaling_dataset_ridotto_dati = standard_scaling_dataset_ridotto.
 ↪fit_transform(dataset_ridotto_Feature_numeriche)
standard_scaling_dataset_ridotto_numerico = pd.
 ↪DataFrame(standard_scaling_dataset_ridotto_dati,␣
 ↪columns=dataset_ridotto_Feature_numeriche.columns)

standard_scaling_dataset_ridotto_numerico
```

[ ]:

	salary
0	-2.226773
1	-2.304900
2	1.110097
3	-0.386460
4	-0.193748

```
..        …
854  0.412167
855  1.792402
856 -0.412502
857 -0.490628
858 -1.185086

[859 rows x 1 columns]
```

[ ]:
```python
import matplotlib.pyplot as plt

colori=["blue"]

# Numero di colonne nel DataFrame
num_cols_dataset_ridotto = len(standard_scaling_dataset_ridotto_numerico.
 ↪columns)

# Creazione dei subplot per gli istogrammi
fig, axes = plt.subplots(nrows=1, ncols=num_cols_dataset_ridotto, figsize=(15,␣
 ↪5))

# Se c'è solo una colonna, axes non sarà una lista, quindi lo mettiamo in una␣
 ↪lista per iterare comunque
if num_cols_dataset_ridotto == 1:
    axes = [axes]

# Loop attraverso le colonne per disegnare gli istogrammi
for i, col in enumerate(standard_scaling_dataset_ridotto_numerico.columns):
    axes[i].hist(standard_scaling_dataset_ridotto_numerico[col], bins=20,␣
 ↪alpha=0.7, color=colori)
    axes[i].set_title('Distribuzione scaling della Feature Salary')   # Imposta␣
 ↪il titolo
    axes[i].set_xlabel('Salario standardizzato')    # Imposta l'etichetta␣
 ↪sull'asse x
    axes[i].set_ylabel('Numero di persone')         # Imposta l'etichetta␣
 ↪sull'asse y

plt.tight_layout()
plt.show()
```



1.8.4 IL ROBUST SCALING

Il Robust scaling scala i dati in modo che possano essere confrontati tra di loro senza essere influenzati da Outliers, questo può essere utile quando nel Dataset a cui si sta lavorando esistono degli Outliers che però non sono stati precedemente eliminati o gestiti. Il Robust scaling quindi riesce a scalare i dati senza che gli Outliers presenti possano "sballare" lo scaling, come invece sarebbe successo nei casi precedenti con le altre tipologie di scaling se non si gestisce prima gli Outliers presenti nel Dataset

[ ]:
```
# Robust scaling
robust_scaling = RobustScaler()
robust_scaling_dati = robust_scaling.fit_transform(dataset_Feature_numeriche)
robust_scaling_dataset_numerico = pd.DataFrame(robust_scaling_dati,␣
 ↪columns=dataset_Feature_numeriche.columns)

robust_scaling_dataset_numerico
```

[ ]:
```
        salary
0    -0.718424
1    -1.297798
2    -1.349942
3     0.382387
4    -0.254925
…          …
1044  0.463499
1045  1.384705
1046 -0.086906
1047 -0.139050
1048 -0.602549

[1049 rows x 1 columns]
```

1.8.5 L'ENCODING ONE HOT

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_encoding = pd.get_dummies(dataset, columns=['experience_level'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_encoding
```

[ ]:
```
                      job_title  salary company_location  experience_level_EN  \
0      Principal Data Scientist   80000               ES                    0
1                   ML Engineer   30000               US                    0
2                   ML Engineer   25500               US                    0
3                Data Scientist  175000               CA                    0
4                Data Scientist  120000               CA                    0
…                           …     …                 …                    …
1809              Data Engineer  182000               US                    0
1814  Machine Learning Engineer  261500               US                    0
1815  Machine Learning Engineer  134500               US                    0
1817             Data Scientist  130000               US                    0
1818             Data Scientist   90000               US                    0

      experience_level_EX  experience_level_MI  experience_level_SE
0                       0                    0                    1
1                       0                    1                    0
2                       0                    1                    0
3                       0                    0                    1
4                       0                    0                    1
…                     …                    …                    …
1809                    0                    0                    1
1814                    0                    0                    1
1815                    0                    0                    1
1817                    0                    1                    0
1818                    0                    1                    0

[1049 rows x 7 columns]
```

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_encoding = pd.get_dummies(dataset, columns=['job_title'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_encoding
```

[ ]:
```
     experience_level  salary company_location  job_title_AI Developer  \
0                  SE   80000               ES                       0
1                  MI   30000               US                       0
2                  MI   25500               US                       0
3                  SE  175000               CA                       0
4                  SE  120000               CA                       0
```

```
…                …     …               …                        …
1809               SE  182000              US                       0
1814               SE  261500              US                       0
1815               SE  134500              US                       0
1817               MI  130000              US                       0
1818               MI   90000              US                       0

      job_title_AI Programmer  job_title_AI Scientist  \
0                           0                        0
1                           0                        0
2                           0                        0
3                           0                        0
4                           0                        0
…                         …                        …
1809                        0                        0
1814                        0                        0
1815                        0                        0
1817                        0                        0
1818                        0                        0

      job_title_Analytics Engineer  job_title_Applied Data Scientist  \
0                                0                                 0
1                                0                                 0
2                                0                                 0
3                                0                                 0
4                                0                                 0
…                              …                                 …
1809                             0                                 0
1814                             0                                 0
1815                             0                                 0
1817                             0                                 0
1818                             0                                 0

      job_title_Applied Machine Learning Engineer  \
0                                               0
1                                               0
2                                               0
3                                               0
4                                               0
…                                             …
1809                                            0
1814                                            0
1815                                            0
1817                                            0
1818                                            0

      job_title_Applied Machine Learning Scientist  …  \
```

```
0                                                    0  …
1                                                    0  …
2                                                    0  …
3                                                    0  …
4                                                    0  …
…                                                  …  …
1809                                                 0  …
1814                                                 0  …
1815                                                 0  …
1817                                                 0  …
1818                                                 0  …

      job_title_Machine Learning Infrastructure Engineer  \
0                                                        0
1                                                        0
2                                                        0
3                                                        0
4                                                        0
…                                                      …
1809                                                     0
1814                                                     0
1815                                                     0
1817                                                     0
1818                                                     0

      job_title_Machine Learning Research Engineer  \
0                                                  0
1                                                  0
2                                                  0
3                                                  0
4                                                  0
…                                                  …
1809                                               0
1814                                               0
1815                                               0
1817                                               0
1818                                               0

      job_title_Machine Learning Researcher  \
0                                         0
1                                         0
2                                         0
3                                         0
4                                         0
…                                         …
1809                                      0
1814                                      0
```

```
1815                                    0
1817                                    0
1818                                    0

      job_title_Machine Learning Scientist  \
0                                        0
1                                        0
2                                        0
3                                        0
4                                        0
…                                      …
1809                                     0
1814                                     0
1815                                     0
1817                                     0
1818                                     0

      job_title_Machine Learning Software Engineer  job_title_NLP Engineer  \
0                                                 0                       0
1                                                 0                       0
2                                                 0                       0
3                                                 0                       0
4                                                 0                       0
…                                               …                       …
1809                                              0                       0
1814                                              0                       0
1815                                              0                       0
1817                                              0                       0
1818                                              0                       0

      job_title_Principal Data Scientist  job_title_Research Engineer  \
0                                      1                            0
1                                      0                            0
2                                      0                            0
3                                      0                            0
4                                      0                            0
…                                    …                            …
1809                                   0                            0
1814                                   0                            0
1815                                   0                            0
1817                                   0                            0
1818                                   0                            0

      job_title_Research Scientist  job_title_Software Data Engineer
0                                0                                 0
1                                0                                 0
2                                0                                 0
```

```
3                                  0                                  0
4                                  0                                  0
…                                  …                                  …
1809                               0                                  0
1814                               0                                  0
1815                               0                                  0
1817                               0                                  0
1818                               0                                  0

[1049 rows x 66 columns]
```

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_encoding = pd.get_dummies(dataset, columns=['company_location'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_encoding
```

[ ]:
```
     experience_level                 job_title  salary  company_location_AM  \
0                  SE   Principal Data Scientist   80000                    0
1                  MI               ML Engineer   30000                    0
2                  MI               ML Engineer   25500                    0
3                  SE             Data Scientist  175000                    0
4                  SE             Data Scientist  120000                    0
…                 …                          …     …                       …
1809               SE              Data Engineer  182000                    0
1814               SE  Machine Learning Engineer  261500                    0
1815               SE  Machine Learning Engineer  134500                    0
1817               MI             Data Scientist  130000                    0
1818               MI             Data Scientist   90000                    0

      company_location_AU  company_location_BA  company_location_BR  \
0                       0                    0                    0
1                       0                    0                    0
2                       0                    0                    0
3                       0                    0                    0
4                       0                    0                    0
…                      …                    …                    …
1809                    0                    0                    0
1814                    0                    0                    0
1815                    0                    0                    0
1817                    0                    0                    0
1818                    0                    0                    0

      company_location_CA  company_location_CF  company_location_CH  …  \
0                       0                    0                    0  …
1                       0                    0                    0  …
2                       0                    0                    0  …
```

```
3                       1                       0                       0  …
4                       1                       0                       0  …
…                     …                       …                     …  …
1809                    0                       0                       0  …
1814                    0                       0                       0  …
1815                    0                       0                       0  …
1817                    0                       0                       0  …
1818                    0                       0                       0  …

      company_location_NG  company_location_NL  company_location_PT  \
0                       0                       0                       0
1                       0                       0                       0
2                       0                       0                       0
3                       0                       0                       0
4                       0                       0                       0
…                     …                       …                     …
1809                    0                       0                       0
1814                    0                       0                       0
1815                    0                       0                       0
1817                    0                       0                       0
1818                    0                       0                       0

      company_location_RO  company_location_SE  company_location_SG  \
0                       0                       0                       0
1                       0                       0                       0
2                       0                       0                       0
3                       0                       0                       0
4                       0                       0                       0
…                     …                       …                     …
1809                    0                       0                       0
1814                    0                       0                       0
1815                    0                       0                       0
1817                    0                       0                       0
1818                    0                       0                       0

      company_location_SI  company_location_UA  company_location_US  \
0                       0                       0                       0
1                       0                       0                       1
2                       0                       0                       1
3                       0                       0                       0
4                       0                       0                       0
…                     …                       …                     …
1809                    0                       0                       1
1814                    0                       0                       1
1815                    0                       0                       1
1817                    0                       0                       1
1818                    0                       0                       1
```

```
      company_location_VN
0                       0
1                       0
2                       0
3                       0
4                       0
…                     …
1809                    0
1814                    0
1815                    0
1817                    0
1818                    0

[1049 rows x 36 columns]
```

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_ridotto_encoding = pd.get_dummies(dataset_ridotto,␣
 ↪columns=['experience_level'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_ridotto_encoding
```

[ ]:
```
                     job_title  salary company_location  experience_level_EN  \
1                   ML Engineer   30000               US                    0
2                   ML Engineer   25500               US                    0
5             Applied Scientist  222200               US                    0
6             Applied Scientist  136000               US                    0
9                Data Scientist  147100               US                    0
…                            …     …                …                     …
1809              Data Engineer  182000               US                    0
1814  Machine Learning Engineer  261500               US                    0
1815  Machine Learning Engineer  134500               US                    0
1817             Data Scientist  130000               US                    0
1818             Data Scientist   90000               US                    0

      experience_level_EX  experience_level_MI  experience_level_SE
1                       0                    1                    0
2                       0                    1                    0
5                       0                    0                    1
6                       0                    0                    1
9                       0                    0                    1
…                     …                    …                    …
1809                    0                    0                    1
1814                    0                    0                    1
1815                    0                    0                    1
1817                    0                    1                    0
```

```
1818                    0                     1                     0

[859 rows x 7 columns]
```

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_ridotto_encoding = pd.get_dummies(dataset_ridotto,␣
 ↪columns=['job_title'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_ridotto_encoding
```

[ ]:
```
     experience_level  salary company_location  job_title_AI Developer  \
1                  MI   30000               US                       0
2                  MI   25500               US                       0
5                  SE  222200               US                       0
6                  SE  136000               US                       0
9                  SE  147100               US                       0
…                 …     …                 …                      …
1809               SE  182000               US                       0
1814               SE  261500               US                       0
1815               SE  134500               US                       0
1817               MI  130000               US                       0
1818               MI   90000               US                       0

      job_title_Analytics Engineer  \
1                                 0
2                                 0
5                                 0
6                                 0
9                                 0
…                                 …
1809                              0
1814                              0
1815                              0
1817                              0
1818                              0

      job_title_Applied Machine Learning Engineer  \
1                                               0
2                                               0
5                                               0
6                                               0
9                                               0
…                                               …
1809                                            0
1814                                            0
1815                                            0
```

```
1817                                          0
1818                                          0

      job_title_Applied Machine Learning Scientist  \
1                                                0
2                                                0
5                                                0
6                                                0
9                                                0
…                                              …
1809                                             0
1814                                             0
1815                                             0
1817                                             0
1818                                             0

      job_title_Applied Scientist  job_title_BI Analyst  \
1                               0                     0
2                               0                     0
5                               1                     0
6                               1                     0
9                               0                     0
…                             …                     …
1809                            0                     0
1814                            0                     0
1815                            0                     0
1817                            0                     0
1818                            0                     0

      job_title_BI Data Engineer  …  job_title_Lead Data Analyst  \
1                              0  …                            0
2                              0  …                            0
5                              0  …                            0
6                              0  …                            0
9                              0  …                            0
…                            …  …                          …
1809                           0  …                            0
1814                           0  …                            0
1815                           0  …                            0
1817                           0  …                            0
1818                           0  …                            0

      job_title_ML Engineer  job_title_MLOps Engineer  \
1                         1                         0
2                         1                         0
5                         0                         0
6                         0                         0
```

```
9                          0                          0
…                        …                          …
1809                       0                          0
1814                       0                          0
1815                       0                          0
1817                       0                          0
1818                       0                          0

      job_title_Machine Learning Engineer  \
1                                       0
2                                       0
5                                       0
6                                       0
9                                       0
…                                     …
1809                                    0
1814                                    1
1815                                    1
1817                                    0
1818                                    0

      job_title_Machine Learning Infrastructure Engineer  \
1                                                       0
2                                                       0
5                                                       0
6                                                       0
9                                                       0
…                                                     …
1809                                                    0
1814                                                    0
1815                                                    0
1817                                                    0
1818                                                    0

      job_title_Machine Learning Scientist  \
1                                        0
2                                        0
5                                        0
6                                        0
9                                        0
…                                      …
1809                                     0
1814                                     0
1815                                     0
1817                                     0
1818                                     0
```

```
      job_title_Machine Learning Software Engineer  job_title_NLP Engineer  \
1                                                0                       0
2                                                0                       0
5                                                0                       0
6                                                0                       0
9                                                0                       0
…                                              …                       …
1809                                             0                       0
1814                                             0                       0
1815                                             0                       0
1817                                             0                       0
1818                                             0                       0

      job_title_Research Engineer  job_title_Research Scientist
1                               0                             0
2                               0                             0
5                               0                             0
6                               0                             0
9                               0                             0
…                             …                             …
1809                            0                             0
1814                            0                             0
1815                            0                             0
1817                            0                             0
1818                            0                             0

[859 rows x 47 columns]
```

[ ]:
```
# Applichiamo l'encoding One-Hot
dataset_ridotto_encoding = pd.get_dummies(dataset_ridotto,␣
 ↪columns=['company_location'])

# Visualizziamo il DataFrame dopo l'encoding
dataset_ridotto_encoding
```

[ ]:
```
     experience_level                job_title  salary  company_location_US
1                  MI               ML Engineer   30000                    1
2                  MI               ML Engineer   25500                    1
5                  SE         Applied Scientist  222200                    1
6                  SE         Applied Scientist  136000                    1
9                  SE            Data Scientist  147100                    1
…                 …                          …     …                       …
1809               SE             Data Engineer  182000                    1
1814               SE  Machine Learning Engineer  261500                    1
1815               SE  Machine Learning Engineer  134500                    1
1817               MI            Data Scientist  130000                    1
1818               MI            Data Scientist   90000                    1
```

[859 rows x 4 columns]

1.8.6 LO SCALING SALVATO NEL DATASET

[ ]:
```
dataset_ridotto["salary"]=min_max_scaling_dataset_numerico["salary"]
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\1576787027.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset_ridotto["salary"]=min_max_scaling_dataset_numerico["salary"]
```

[ ]:
```
dataset_ridotto
```

[ ]:

	experience_level	job_title	salary	company_location
1	MI	ML Engineer	0.045726	US
2	MI	ML Engineer	0.036779	US
5	SE	Applied Scientist	0.427833	US
6	SE	Applied Scientist	0.256461	US
9	SE	Data Scientist	0.278529	US
…	…	…	…	…
1809	SE	Data Engineer	NaN	US
1814	SE	Machine Learning Engineer	NaN	US
1815	SE	Machine Learning Engineer	NaN	US
1817	MI	Data Scientist	NaN	US
1818	MI	Data Scientist	NaN	US

[859 rows x 4 columns]

[ ]:
```
# Calcolo del totale delle righe con dati mancanti
totale_dati_mancanti_dataset_ridotto = dataset_ridotto.isnull().any(axis=1).
 ↪sum() # Calcola il totale delle righe con almeno un dato mancante

# Determinazione delle colonne con dati mancanti
colonne_dati_mancanti_dataset_ridotto = dataset_ridotto.isnull().any(axis=0) #␣
 ↪True se almeno un valore nella colonna è mancante (None o NaN)
```

[ ]:
```
# Stampa delle colonne con dati mancanti e del totale dei dati mancanti
print("Colonne con i NaN nel Dataset ridotto dopo il Min-Max Scaling:")
print(colonne_dati_mancanti_dataset_ridotto)
print(f"Totale delle righe con i NaN nel Dataset ridotto dopo il Min-Max␣
 ↪Scaling: {totale_dati_mancanti_dataset_ridotto}")
```

```
Colonne con i NaN nel Dataset ridotto dopo il Min-Max Scaling:
```

```
experience_level    False
job_title           False
salary               True
company_location    False
dtype: bool
Totale delle righe con i NaN nel Dataset ridotto dopo il Min-Max Scaling: 287
```

[ ]:
```
# Visualizzare solo i valori mancanti nella feature specificata
dati_mancanti_salary = dataset_ridotto[dataset_ridotto['salary'].isnull()]

# Stampare i valori mancanti della feature specificata
dati_mancanti_salary
```

[ ]:
```
     experience_level                                job_title  salary  \
1054               SE                    Data Science Manager     NaN
1055               SE                    Data Science Manager     NaN
1056               SE                            Data Analyst     NaN
1061               SE                            Data Manager     NaN
1062               SE  Machine Learning Infrastructure Engineer     NaN
…                 …                                       …     …
1809               SE                           Data Engineer     NaN
1814               SE                Machine Learning Engineer     NaN
1815               SE                Machine Learning Engineer     NaN
1817               MI                           Data Scientist     NaN
1818               MI                           Data Scientist     NaN

     company_location
1054               US
1055               US
1056               US
1061               US
1062               US
…                 …
1809               US
1814               US
1815               US
1817               US
1818               US

[287 rows x 4 columns]
```

[ ]:
```
len(dati_mancanti_salary)
```

[ ]:
```
287
```

[ ]:
```
# Riempire i valori mancanti nella colonna 'salary' con la media
dataset_ridotto['salary'] = dataset_ridotto['salary'].
  ↪fillna(dataset_ridotto['salary'].mean(numeric_only=True))
```

```
# Stampare il DataFrame con i valori mancanti corretti
dataset_ridotto
```

```
C:\Users\matte\AppData\Local\Temp\ipykernel_8728\396318002.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset_ridotto['salary'] = dataset_ridotto['salary'].fillna(dataset_ridotto['
salary'].mean(numeric_only=True))
```

[ ]:

	experience_level	job_title	salary	company_location
1	MI	ML Engineer	0.045726	US
2	MI	ML Engineer	0.036779	US
5	SE	Applied Scientist	0.427833	US
6	SE	Applied Scientist	0.256461	US
9	SE	Data Scientist	0.278529	US
…	…	…	…	…
1809	SE	Data Engineer	0.286419	US
1814	SE	Machine Learning Engineer	0.286419	US
1815	SE	Machine Learning Engineer	0.286419	US
1817	MI	Data Scientist	0.286419	US
1818	MI	Data Scientist	0.286419	US

[859 rows x 4 columns]

[ ]:
```
# Visualizzare solo i valori mancanti nella feature specificata
dati_mancanti_salary = dataset_ridotto[dataset_ridotto['salary'].isnull()]

# Stampare i valori mancanti della feature specificata
dati_mancanti_salary
```

[ ]:
```
Empty DataFrame
Columns: [experience_level, job_title, salary, company_location]
Index: []
```

1.9 FASE 9: LO SPLITTING DATASET E I GRAFICI CORRELATI

[ ]:
```
dataset
```

[ ]:

	experience_level	job_title	salary	company_location
0	SE	Principal Data Scientist	80000	ES
1	MI	ML Engineer	30000	US
2	MI	ML Engineer	25500	US
3	SE	Data Scientist	175000	CA

```
4                  SE              Data Scientist  120000               CA
…                 …                           …     …                  …
1809               SE               Data Engineer  182000               US
1814               SE  Machine Learning Engineer  261500               US
1815               SE  Machine Learning Engineer  134500               US
1817               MI              Data Scientist  130000               US
1818               MI              Data Scientist   90000               US

[1049 rows x 4 columns]
```

[ ]:
```
dataset_ridotto
```

[ ]:
```
     experience_level                 job_title    salary company_location
1                  MI                ML Engineer  0.045726               US
2                  MI                ML Engineer  0.036779               US
5                  SE           Applied Scientist  0.427833               US
6                  SE           Applied Scientist  0.256461               US
9                  SE              Data Scientist  0.278529               US
…                 …                           …       …                  …
1809               SE               Data Engineer  0.286419               US
1814               SE  Machine Learning Engineer  0.286419               US
1815               SE  Machine Learning Engineer  0.286419               US
1817               MI              Data Scientist  0.286419               US
1818               MI              Data Scientist  0.286419               US

[859 rows x 4 columns]
```

[ ]:
```
dataset=dataset_ridotto
```

[ ]:
```
dataset
```

[ ]:
```
     experience_level                 job_title    salary company_location
1                  MI                ML Engineer  0.045726               US
2                  MI                ML Engineer  0.036779               US
5                  SE           Applied Scientist  0.427833               US
6                  SE           Applied Scientist  0.256461               US
9                  SE              Data Scientist  0.278529               US
…                 …                           …       …                  …
1809               SE               Data Engineer  0.286419               US
1814               SE  Machine Learning Engineer  0.286419               US
1815               SE  Machine Learning Engineer  0.286419               US
1817               MI              Data Scientist  0.286419               US
1818               MI              Data Scientist  0.286419               US

[859 rows x 4 columns]
```

[ ]:
```
import numpy as np
from sklearn.model_selection import train_test_split # in questo caso viene␣
 ↪solo importata una parte di libreria poichè è strettamente necessaria quella␣
 ↪determinata funzione

valori_salary = dataset["salary"]
valori_salary
```

[ ]:
```
1       0.045726
2       0.036779
5       0.427833
6       0.256461
9       0.278529
          …
1809    0.286419
1814    0.286419
1815    0.286419
1817    0.286419
1818    0.286419
Name: salary, Length: 859, dtype: float64
```

[ ]:
```
valori_job_title = dataset["job_title"]
valori_job_title
```

[ ]:
```
1                     ML Engineer
2                     ML Engineer
5                Applied Scientist
6                Applied Scientist
9                   Data Scientist
                      …
1809                 Data Engineer
1814    Machine Learning Engineer
1815    Machine Learning Engineer
1817                Data Scientist
1818                Data Scientist
Name: job_title, Length: 859, dtype: object
```

[ ]:
```
# Suddividere il dataset in training set (70%) e test set (30%) formando due␣
 ↪DataSet
X_train, X_test, y_train, y_test = train_test_split(valori_salary,␣
 ↪valori_job_title, test_size=0.3, random_state=42) # la formula è: le X sono␣
 ↪i valori del salary perchè sono le Feature del DataSet, cioè l'input. Invece␣
 ↪le Y sono gli output o target del DataSet, cioè i valori del job title.␣
 ↪"test_size=0.3" vuol dire che il DataSet di Test è il 30% di quello totale␣
 ↪mentre random_state sceglie in modo randomico i valori del DataSet per il␣
 ↪Training e il Test
# Stampare le dimensioni dei training set e test set
```

```
print("Dimensioni del Training Set (valori di \"salary\" e valori␣
 ↪\"job_title\"):", X_train.shape, y_train.shape) # shape = dimensione dei␣
 ↪DataSet di Training
print("Dimensioni del Test Set (valori di \"salary\" e valori \"job_title\"):",␣
 ↪X_test.shape, y_test.shape) # shape = dimensione dei DataSet di Test
```

```
Dimensioni del Training Set (valori di "salary" e valori "job_title"): (601,)
(601,)
Dimensioni del Test Set (valori di "salary" e valori "job_title"): (258,) (258,)
```

[ ]:
```
import matplotlib.pyplot as plt

# Visualizzare le distribuzioni dei valori di 'salary' nel training set e nel␣
 ↪test set
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.hist(X_train, bins=20, color='skyblue', alpha=0.7)
plt.title('Distribuzione di Salary nel Training Set')
plt.xlabel('Salary')
plt.ylabel('Frequenza')

plt.subplot(1, 2, 2)
plt.hist(X_test, bins=20, color='salmon', alpha=0.7)
plt.title('Distribuzione di Salary nel Test Set')
plt.xlabel('Salary')
plt.ylabel('Frequenza')

plt.tight_layout()
plt.show()
```

[ ]:
```
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.bar(y_train.value_counts().index, y_train.value_counts().values,␣
 ↪color='skyblue')
plt.title('Distribuzione di Job Title nel Training Set')
plt.xlabel('Job Title')
plt.ylabel('Numero di Occorrenze')
plt.xticks(rotation=90, ha='right')

plt.subplot(1, 2, 2)
plt.bar(y_test.value_counts().index, y_test.value_counts().values,␣
 ↪color='salmon')
plt.title('Distribuzione di Job Title nel Test Set')
plt.xlabel('Job Title')
plt.ylabel('Numero di Occorrenze')

plt.xticks(rotation=90, ha='right')
plt.tight_layout()
plt.show()
```

1.10 FASE 10: L'ANALISI REALISTICA FINALE DEL DATASET (CON TUTTE LE MODIFICHE EFFETUATE PRECEDENTEMENTE)

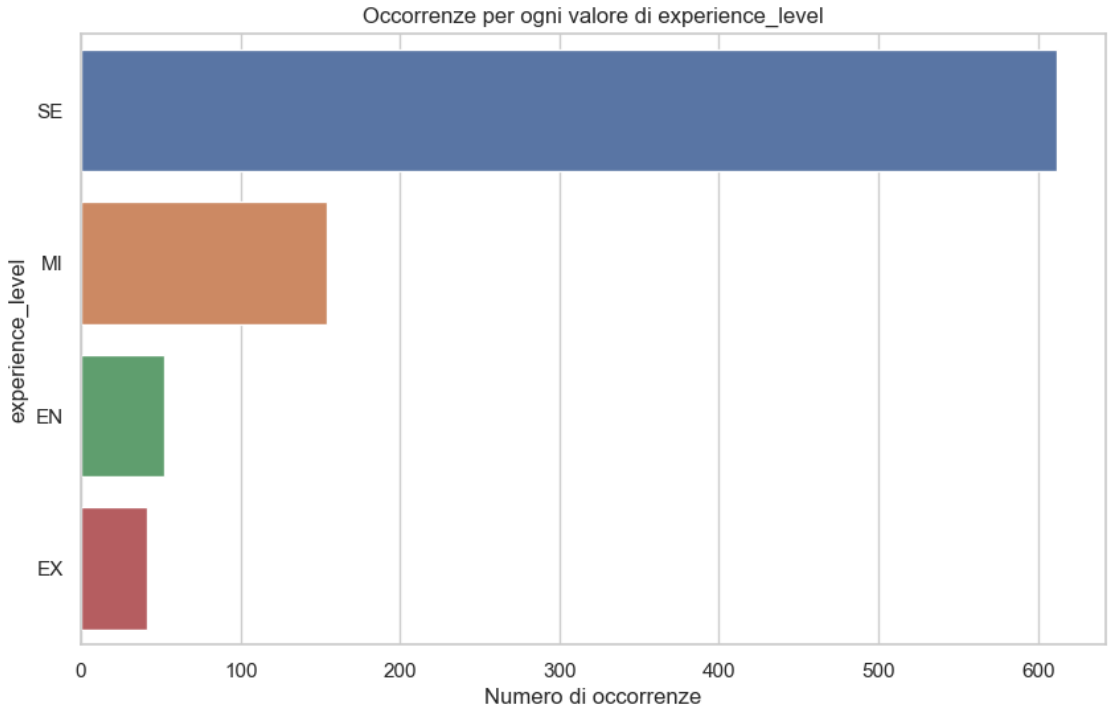
[ ]:
```
dataset
```

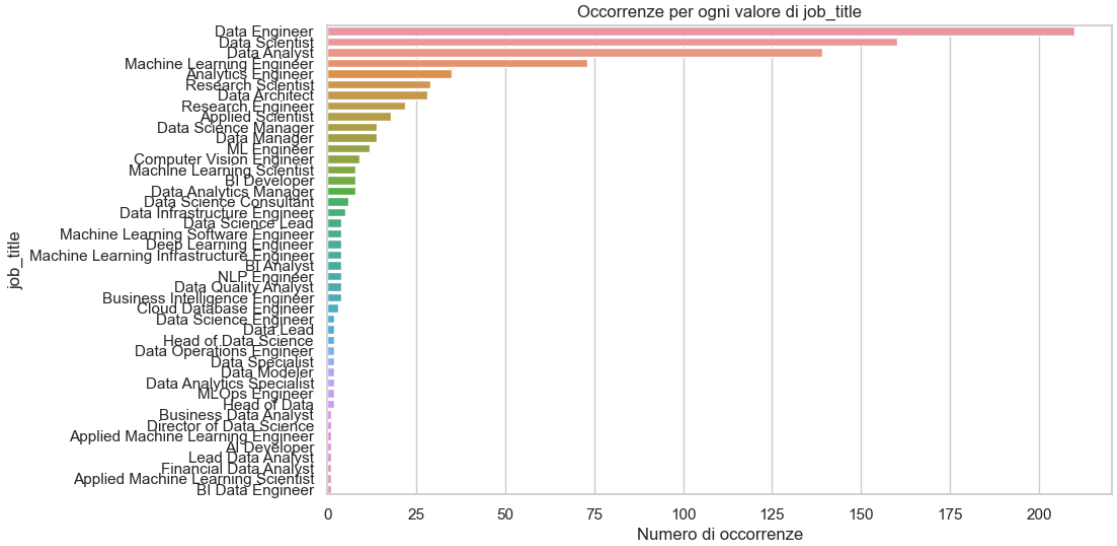
[ ]:

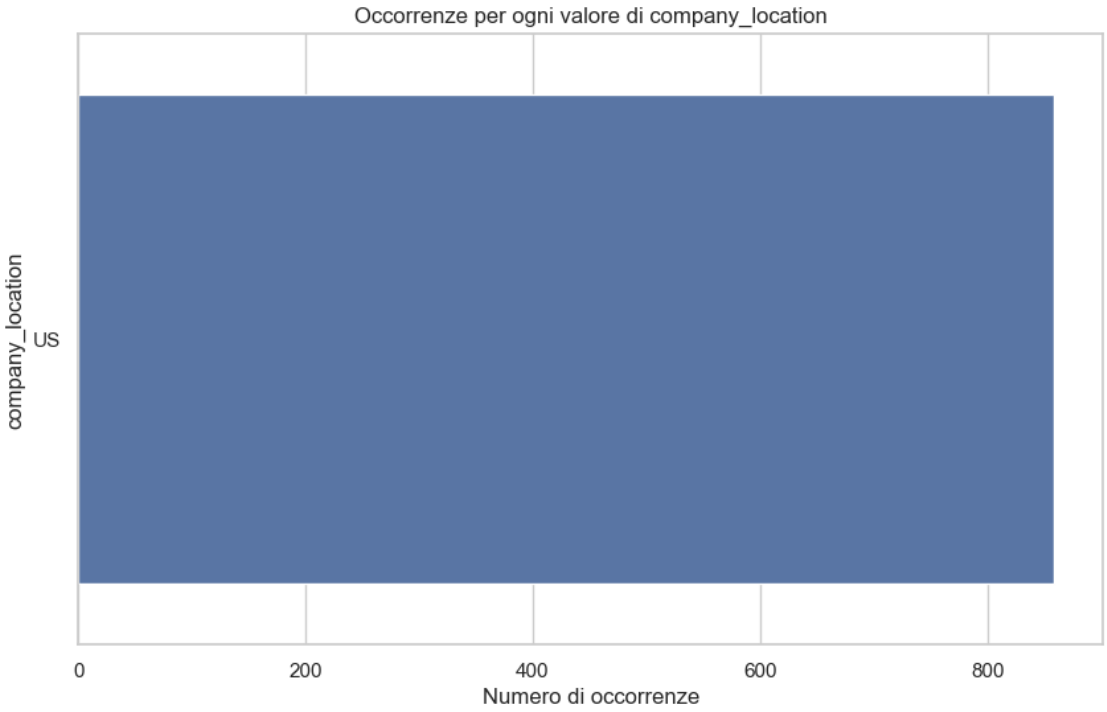
	experience_level	job_title	salary	company_location
1	MI	ML Engineer	0.045726	US
2	MI	ML Engineer	0.036779	US

```
5                  SE          Applied Scientist  0.427833              US
6                  SE          Applied Scientist  0.256461              US
9                  SE             Data Scientist  0.278529              US
…                 …                          …        …                …
1809               SE              Data Engineer  0.286419              US
1814               SE  Machine Learning Engineer  0.286419              US
1815               SE  Machine Learning Engineer  0.286419              US
1817               MI             Data Scientist  0.286419              US
1818               MI             Data Scientist  0.286419              US

[859 rows x 4 columns]
```

[ ]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Impostazione dello stile di visualizzazione per i grafici
sns.set(style="whitegrid")

# Definizione delle feature per le quali si vuole fare il grafico delle␣
 ↪occorrenze
features = ["experience_level", "job_title", "company_location"]

# Creazione dei grafici per ogni feature
for feature in features:
    plt.figure(figsize=(10, 6))  # Imposta le dimensioni del grafico
    sns.countplot(y=feature, data=dataset, order = dataset[feature].
 ↪value_counts().index)  # Crea il grafico a barre
    plt.title(f'Occorrenze per ogni valore di {feature}')  # Titolo del grafico
    plt.xlabel('Numero di occorrenze')  # Etichetta asse x
    plt.ylabel(feature)  # Etichetta asse y
    plt.show()  # Mostra il grafico
```


Occorrenze per ogni valore di experience_level
SE
MI
EN
EX
experience_level
0
100
200
300
400
500
600
Numero di occorrenze


Occorrenze per ogni valore di job_title
Data Engineer
Data Scientist
Data Analyst
Machine Learning Engineer
Analytics Engineer
Research Scientist
Data Architect
Research Engineer
Applied Scientist
Data Science Manager
Data Manager
ML Engineer
Computer Vision Engineer
Machine Learning Scientist
BI Developer
Data Analytics Manager
Data Science Consultant
Data Infrastructure Engineer
Data Science Lead
Machine Learning Software Engineer
Deep Learning Engineer
Machine Learning Infrastructure Engineer
BI Analyst
NLP Engineer
Data Quality Analyst
Business Intelligence Engineer
Cloud Database Engineer
Data Science Engineer
Data Lead
Head of Data Science
Data Operations Engineer
Data Specialist
Data Modeler
Data Analytics Specialist
MLOps Engineer
Head of Data
Business Data Analyst
Director of Data Science
Applied Machine Learning Engineer
AI Developer
Lead Data Analyst
Financial Data Analyst
Applied Machine Learning Scientist
BI Data Engineer
job_title
0
25
50
75
100
125
150
175
200
Numero di occorrenze



[ ]:
```
import pandas as pd

valori_maggiori_dataset = dataset['salary'].nlargest(10)

valori_maggiori_dataset
```

[ ]:
```
79     1.000000
480    0.733757
721    0.721670
127    0.666600
197    0.641153
139    0.618887
376    0.616441
857    0.610537
612    0.602386
67     0.601193
Name: salary, dtype: float64
```