MINIPROJECT - Alice, Caitlin, Jamil, Kristof & Valeria

1. Answer the questions

- **What are the main desired profiles for candidates?**
  We ran a python model aiming at identifying and counting the **most recurring words** in the "Position" column, based on **seniority** and **actual role**. Of the 41% identified in the seniority column (Table 1), the majority are relatively advanced (e.g. "senior", "manager", "principal", …). Of the 60% identified for the role, "scientist", "engineer" and "analyst" were the most common (Table 2).

- **Which location has the most opportunities?**
  New York offers the most opportunities for Data Scientist job hunters with 848 open job positions (Table 3), followed by Seattle, Cambridge, Boston and San Francisco.

2. The Process

- We started with analysing the **null values**. In most columns, there are only 11 rows that have null values (Table 4). We concluded that these 11 rows are likely entirely null values, but even if they are not, the number is still very small vis-a-vis the overall data frame. We decided to drop those rows. We then focused on the reviews[1] (Table 5), which have a 24% null-value-rate and we felt the information in this column contributed little. We rejected the main methodologies[2] approaching the missing values and just decided to drop the review column.
- We employed two methods to use **Boolean operators** to indicate which roles are closest to big centres in the U.S. Both were successful[3], but would need some tweaking to be better. The challenge was cleaning the location data and separating the city, state, and zip codes out for each other. In both cases, we then created a list of cities that were identified as being 'hubs' and then used *map* with a *lambda* function to identify the matches between our new 'locations' column and the hub list and populate a new column with the boolean answer (Table 6).

3. Challenges & Next Steps

- Firstly, we can push the analysis further on the "position" column by **binning** the "seniority" and "role" data that we collected. For example, grouping seniority levels by "senior", "middle", "junior".
- Secondly, we need to apply our model to the "**description**" column to identify the most sought after job profiles (areas, year of experience, skill set).
- Finally, the model we used to extract information from the "position" column is very comprehensive in analysing every word. This might represent a **limitation** if applied to a more content-heavy column like "description".
- More generally speaking, reading more documentation on **NLP** would have benefited the outcome of this analysis.

4. Feedback on Group Work & Limitations

- There were challenges with working online. Furthermore, the time pressure was difficult, especially considering the large number of labs and workshops we had this week and the level of difficulty of the content.
- We also had difficulty choosing between working in SQL and Python. Additional difficulties loading the file into MySql Workbench.
- We ended up splitting the work between two mini-groups. In the end, this was effective, but it was difficult to decide how to split up the work. At first, we planned to split it on the SQL-Python axis, but in the end, everyone ended up wanting to work in Python due to what we were learning in class at the time and the opportunity to try to apply the skills we were learning in class to the project. In the end, however, we all enjoyed working with each other!

Table 1.

```
In [61]: df1["role"].value_counts()

Out[61]: scientist      2023
         engineer        920
         analyst         715
         developer       101
         architect        62
         technician       51
         researcher       45
         programmer       42
         consultant       42
         postdoctoral     34
```

Table 2.

```
In [60]: df1["seniority"].value_counts()

Out[60]: None         4108
         senior       1001
         associate     331
         manager       305
         sr.           266
         principal     195
         lead          157
         specialist    123
         assistant     101
         director       92
         sr             92
```

Table 3.

	location
New York, NY	848
Seattle, WA	777
Cambridge, MA	694
Boston, MA	629
San Francisco, CA	564

Table 4.

```
data.isnull().sum()

position          11
company           11
description       11
reviews         1638
location          11
dtype: int64
```

Table 5.

	column	Percentage Null Values
3	reviews	0.23521
0	position	0.00158
1	company	0.00158
2	description	0.00158
4	location	0.00158

Table 6.

	Hub_Proximity
False	4322
True	2631

1. Reviews could be an indicator of how large a company is, how well known the company, or how many people are talking about it. However, since we wanted to focus on what companies were looking for in job candidates and where these opportunities were, we felt the information in this column contributed little.
2. Rejected methodologies: guessing the missing value, replacing the missing value with either 0 or 'unknown', dropping the rows that had missing values, replacing missing values with the median, mean or mode, using linear regression, interpolation, or the random forest of KNN models to fill in the missing values.
3. One used the *str.split* function in Python to create a new data frame with columns for city, state, and zip code and concatenated it with the original dataframe. The other used *str.replace* with regex to identify the numbers and take the zip codes out.