Data Cleaning & Exploration

Preparing and Preprocessing the Job Market Dataset

Connor Coulter

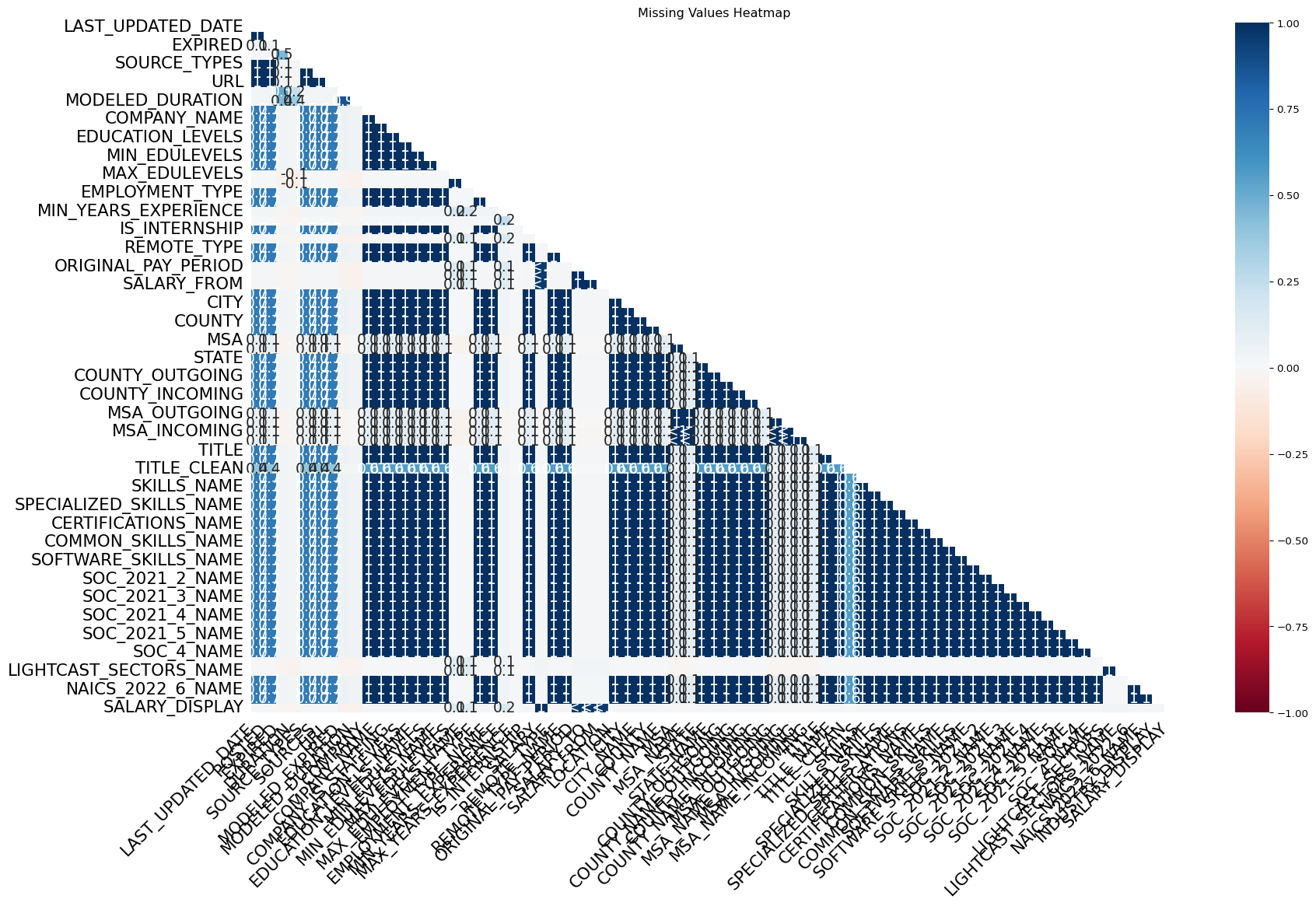
Wei Wang

Balqis Bevi Abdul Hannan Kanaga

# Import Data

import pandas as pd, numpy as np, os, missingno as msno, matplotlib.pyplot as plt  
  
# load data  
CSV\_PATHS = ["data/lightcast\_job\_postings.csv", "lightcast\_job\_postings.csv"]  
csv\_path = next((p for p in CSV\_PATHS if os.path.exists(p)), None)  
if not csv\_path:  
 raise FileNotFoundError("⚠️ lightcast\_job\_postings.csv not found")  
  
df = pd.read\_csv(csv\_path, low\_memory=False)  
print("Loaded dataset:", df.shape)  
  
df["INDUSTRY\_DISPLAY"] = (  
 df["NAICS\_2022\_6\_NAME"]  
 if "NAICS\_2022\_6\_NAME" in df.columns  
 else df.get("INDUSTRY", pd.Series(["Unknown"]\*len(df)))  
)  
  
salary\_candidates = ["SALARY","SALARY\_MEDIAN","SALARY\_MID","SALARY\_ANNUAL","PAY\_RATE"]  
for c in salary\_candidates:  
 if c in df.columns:  
 df[c] = pd.to\_numeric(df[c], errors="coerce")  
  
df["SALARY\_DISPLAY"] = next(  
 (df[c] for c in salary\_candidates if c in df.columns),  
 pd.Series([np.nan]\*len(df))  
)  
  
# drops unused cols  
columns\_to\_drop = [  
 "ID","LAST\_UPDATED\_TIMESTAMP","DUPLICATES","ACTIVE\_URLS","ACTIVE\_SOURCES\_INFO",  
 "TITLE\_RAW","BODY","COMPANY\_RAW",  
 "NAICS2","NAICS2\_NAME","NAICS3","NAICS3\_NAME","NAICS4","NAICS4\_NAME",  
 "NAICS5","NAICS5\_NAME","NAICS6","NAICS6\_NAME",  
 "NAICS\_2022\_2","NAICS\_2022\_2\_NAME","NAICS\_2022\_3","NAICS\_2022\_3\_NAME",  
 "NAICS\_2022\_4","NAICS\_2022\_4\_NAME","NAICS\_2022\_5","NAICS\_2022\_5\_NAME",  
 "SOC\_2","SOC\_2\_NAME","SOC\_3","SOC\_3\_NAME","SOC\_5","SOC\_5\_NAME",  
 "CIP2","CIP2\_NAME","CIP4","CIP4\_NAME","CIP6","CIP6\_NAME",  
 "LOT\_CAREER\_AREA","LOT\_CAREER\_AREA\_NAME","LOT\_OCCUPATION","LOT\_OCCUPATION\_NAME",  
 "LOT\_SPECIALIZED\_OCCUPATION","LOT\_SPECIALIZED\_OCCUPATION\_NAME",  
 "LOT\_OCCUPATION\_GROUP","LOT\_OCCUPATION\_GROUP\_NAME",  
 "LOT\_V6\_SPECIALIZED\_OCCUPATION","LOT\_V6\_SPECIALIZED\_OCCUPATION\_NAME",  
 "LOT\_V6\_OCCUPATION","LOT\_V6\_OCCUPATION\_NAME","LOT\_V6\_OCCUPATION\_GROUP",  
 "LOT\_V6\_OCCUPATION\_GROUP\_NAME","LOT\_V6\_CAREER\_AREA","LOT\_V6\_CAREER\_AREA\_NAME",  
 "ONET","ONET\_NAME","ONET\_2019","ONET\_2019\_NAME"  
]  
drop\_existing = [c for c in columns\_to\_drop if c in df.columns]  
df.drop(columns=drop\_existing, inplace=True)  
print("Remaining columns (first 30):", list(df.columns)[:30])  
  
# handle missing vals  
msno.heatmap(df)  
plt.title("Missing Values Heatmap")  
plt.show()  
  
df.dropna(thresh=len(df) \* 0.5, axis=1, inplace=True)  
  
if "SALARY\_DISPLAY" in df.columns:  
 df["SALARY\_DISPLAY"].fillna(df["SALARY\_DISPLAY"].median(), inplace=True)  
  
for col in df.select\_dtypes(include="object").columns:  
 df[col].fillna("Unknown", inplace=True)  
  
# remove all duplicates  
subset\_cols = [c for c in ["TITLE","COMPANY\_NAME","LOCATION","POSTED"] if c in df.columns]  
if subset\_cols:  
 before = len(df)  
 df.drop\_duplicates(subset=subset\_cols, keep="first", inplace=True)  
 print(f"Removed {before - len(df)} duplicates using {subset\_cols}")

Loaded dataset: (72498, 131)  
Remaining columns (first 30): ['LAST\_UPDATED\_DATE', 'POSTED', 'EXPIRED', 'DURATION', 'SOURCE\_TYPES', 'SOURCES', 'URL', 'MODELED\_EXPIRED', 'MODELED\_DURATION', 'COMPANY', 'COMPANY\_NAME', 'COMPANY\_IS\_STAFFING', 'EDUCATION\_LEVELS', 'EDUCATION\_LEVELS\_NAME', 'MIN\_EDULEVELS', 'MIN\_EDULEVELS\_NAME', 'MAX\_EDULEVELS', 'MAX\_EDULEVELS\_NAME', 'EMPLOYMENT\_TYPE', 'EMPLOYMENT\_TYPE\_NAME', 'MIN\_YEARS\_EXPERIENCE', 'MAX\_YEARS\_EXPERIENCE', 'IS\_INTERNSHIP', 'SALARY', 'REMOTE\_TYPE', 'REMOTE\_TYPE\_NAME', 'ORIGINAL\_PAY\_PERIOD', 'SALARY\_TO', 'SALARY\_FROM', 'LOCATION']



/tmp/ipykernel\_5967/3461735379.py:61: FutureWarning:  
  
A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.

Removed 3300 duplicates using ['TITLE', 'COMPANY\_NAME', 'LOCATION', 'POSTED']