Machine Learning Methods

Regression, Classification, Topic Insights

October 16, 2025

## title: “About”

About this site

# Introduction and Research Rationale

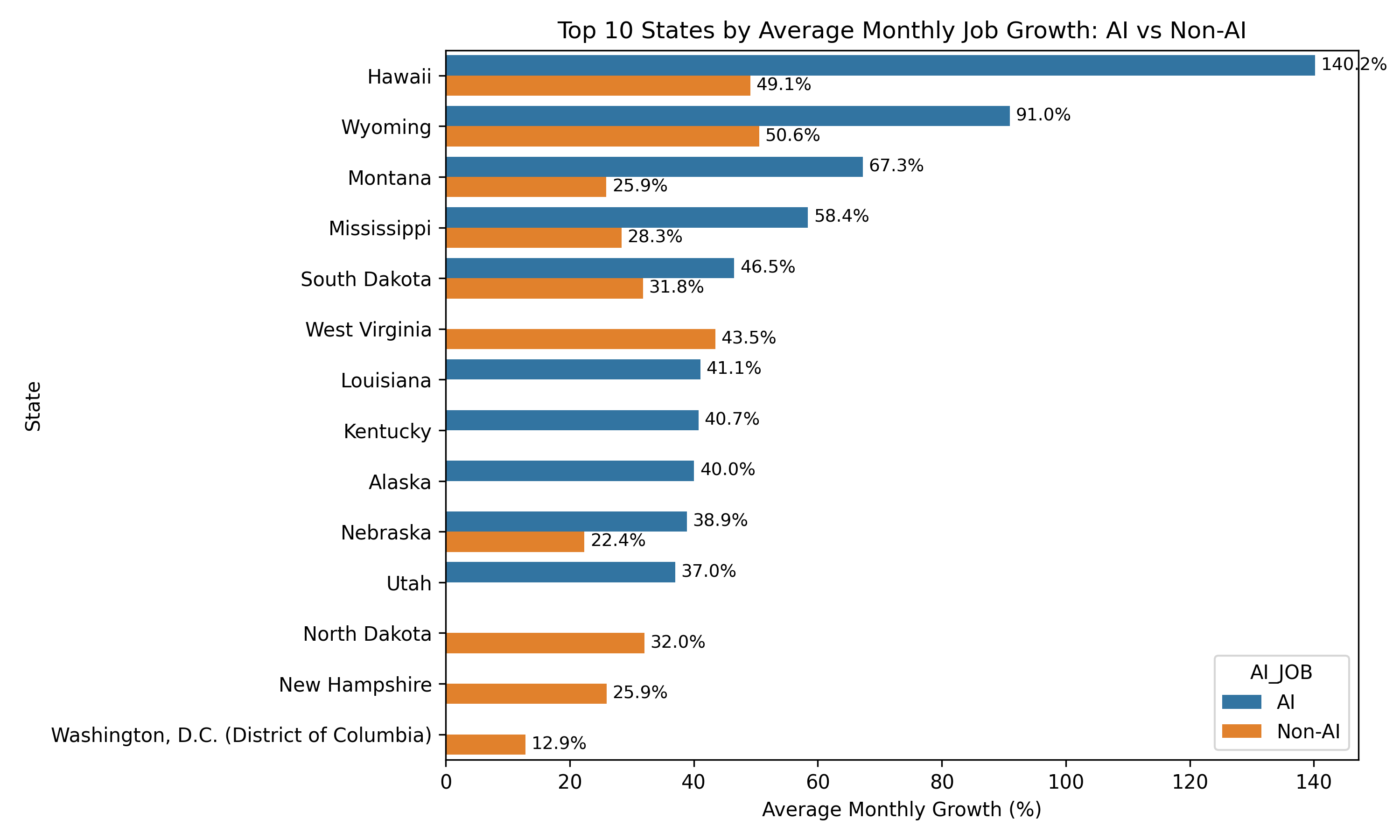
Artificial intelligence, also known as AI, is redefining the labor market, shaping not only how work is done but also where work is done. As businesses adopt AI-driven tools and automation, demand for data science, machine learning, and software roles continues to grow, while routine administrative and operational jobs decline (Kahn et al, 2024). This recent shift has created a new geography of employment, where technological infrastructure and remote work options play central roles.

Remote work, accelerated by the COVID-19 pandemic and continued by companies adoption of this practice, has ereated more high-wage, skill-based positions. However, the benefits of this expansion are unevenly distributed as we will see in our analysis. Traditional technology hubs such as Silicon Valley, Austin, and Boston continue to hire a high number of tech/AI affected roles, while smaller metro areas and rural communities face challenges in attracting and retaining AI-related talent (Sheffi, 2024). At the same time, our data shows that remote and hybrid work models are allowing some decentralization, offering emerging regions new opportunities for growth.

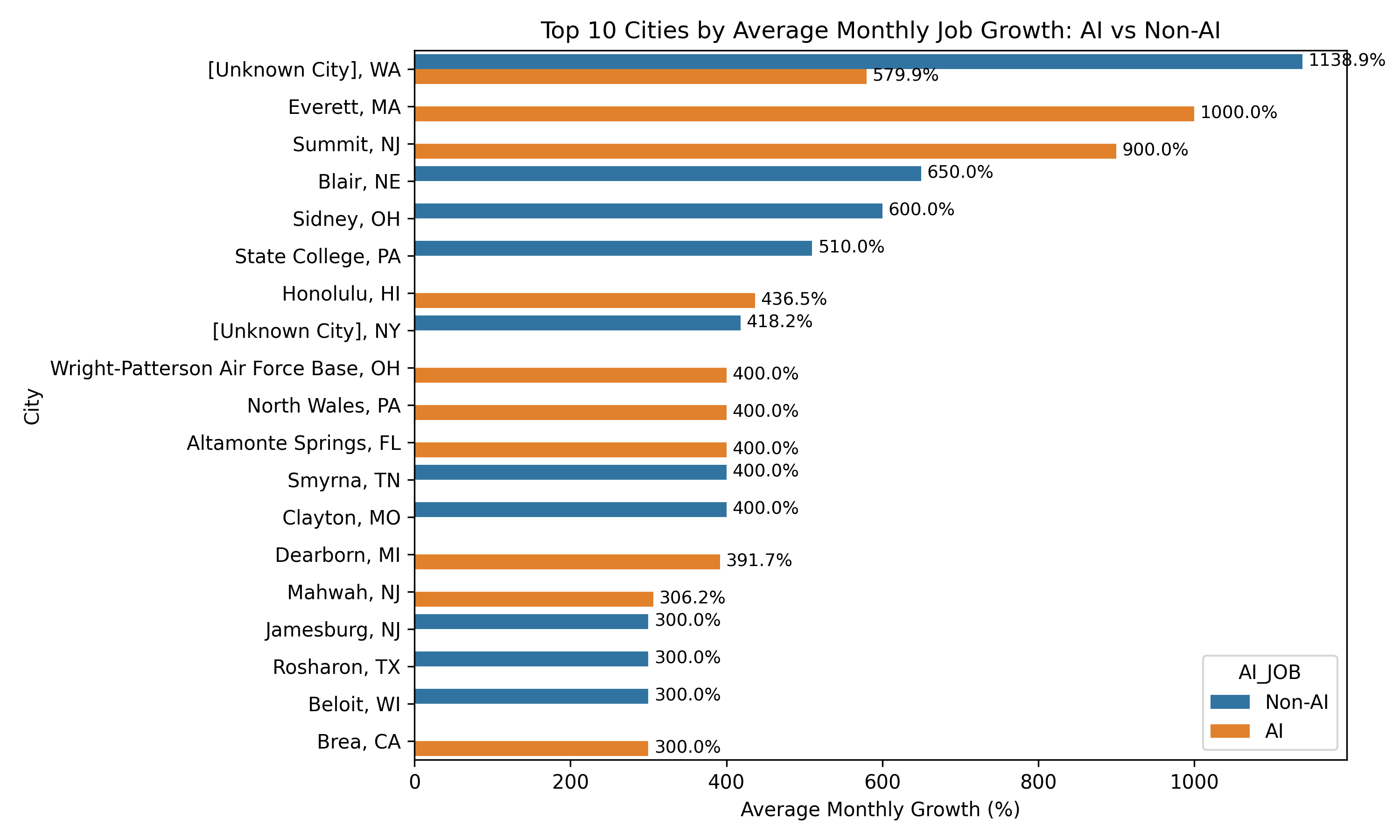
This research analyzes how AI roles and other non-AI roles may be affected by geographic and remote work trends in 2025. The study investigates which states and urban areas demonstrate the highest job growth in AI versus non-AI careers, whether remote positions are increasing across industries, and how urban and rural job markets differ in AI accessibility. Based on our research, it is expected that AI roles will remain concentrated in tech-oriented regions, but it also seems that other non traditional metro areas may be seeing growth not previously seen. Also, remote work is anticipated to remain more prevalent in AI and technology fields, as well as higher salaries in both AI related jobs and remote jobs (Solo et al., 2025). Below are the main focal points of our research:

## AI vs. Non-AI Job Growth

Preliminary findings indicate that AI-related positions show the strongest concentration in technology-intensive states such as California, Washington, and Massachusetts. However, states like Utah, North Carolina, and Colorado show rising growth rates, suggesting decentralization from traditional hubs. In contrast, non-AI jobs remain more evenly distributed, with steady growth across states with large service or manufacturing economies.



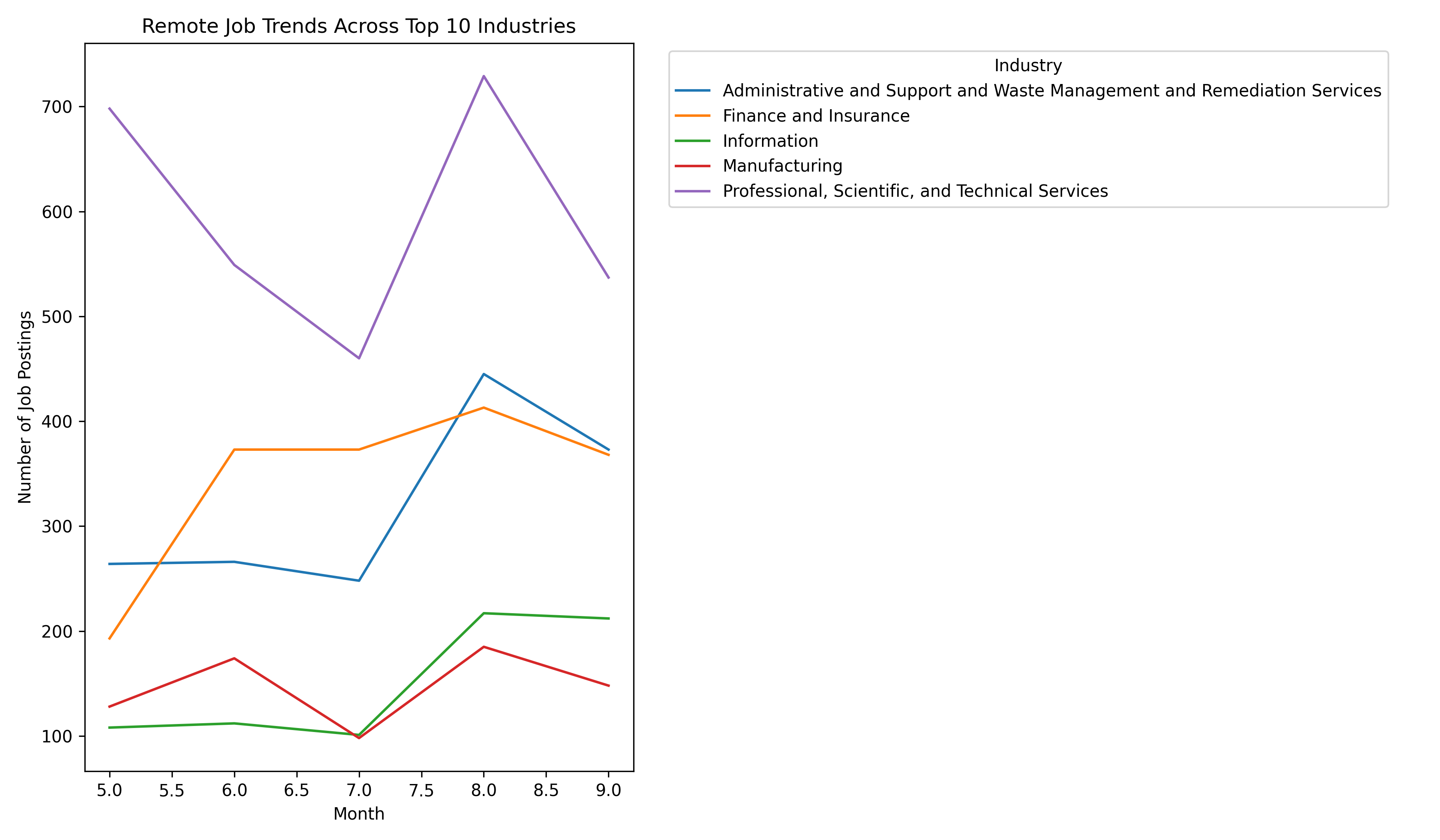
AI vs. Non-AI Job Growth - State



AI vs. Non-AI Job Growth - City

## Trends in Remote Work Across Industries

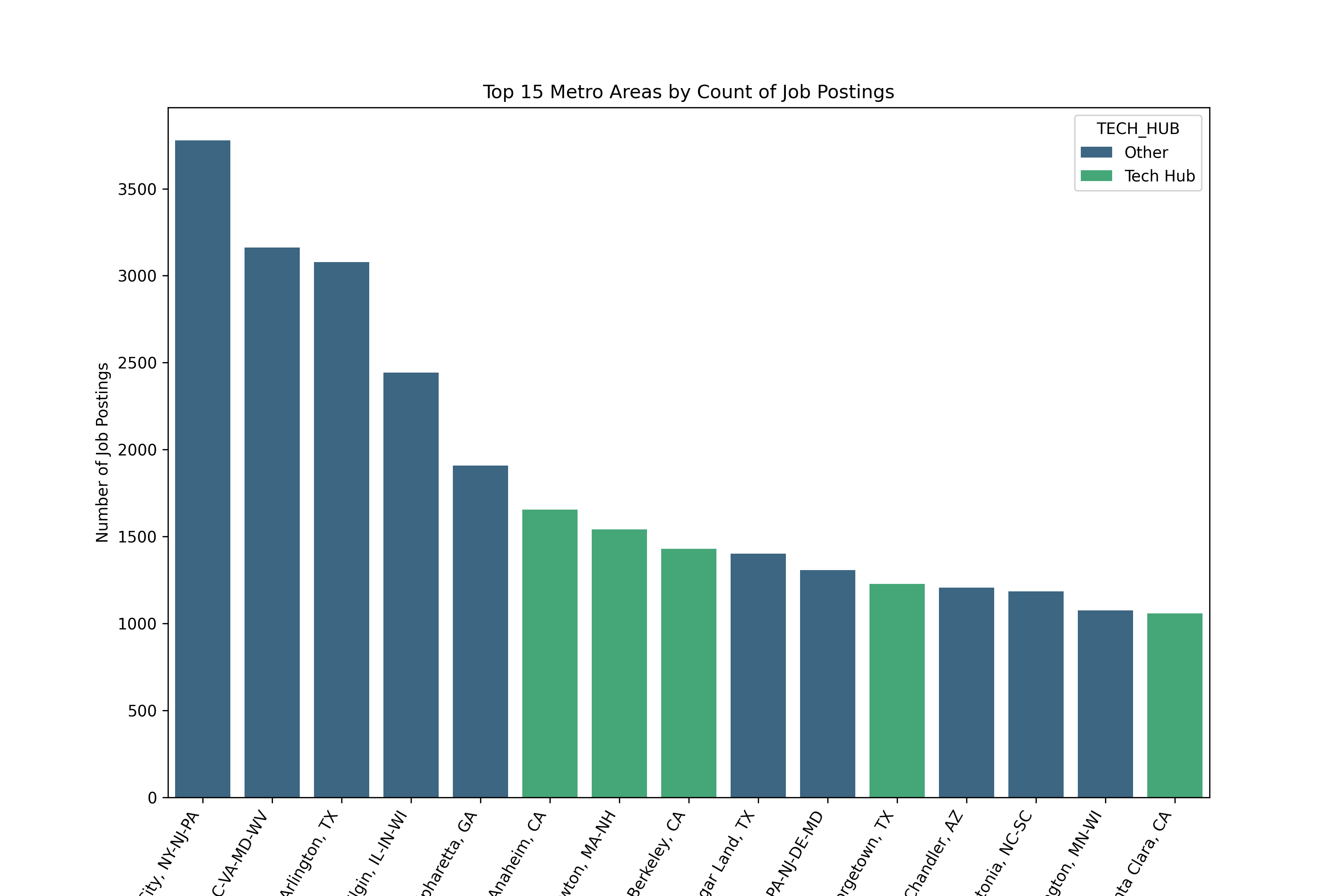
Remote job listings continue to represent a significant share of AI-related careers, especially in data analysis, software engineering, and research roles. Industries such as finance, education technology, and digital media also maintain high remote participation. Conversely, healthcare, logistics, and manufacturing remain predominantly on-site. Overall, remote opportunities appear to be stabilizing rather than growing, signaling a post-pandemic normalization.



Remote Work Trends

## Tech Hubs vs. Emerging Locations

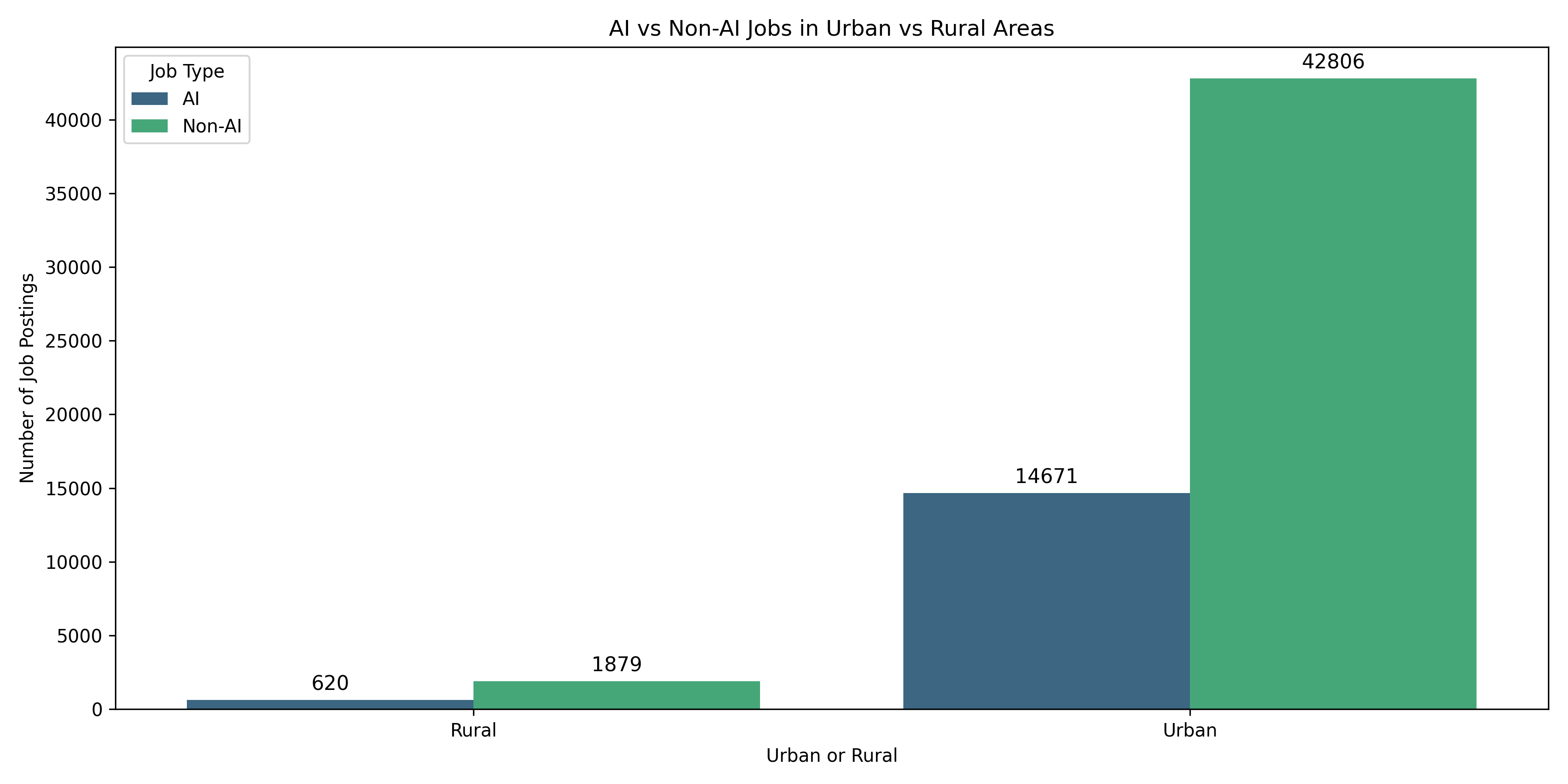
While legacy tech hubs liek Silicon Valley, Austin, Boston, and Seattle still dominate total AI job volume, emerging markets are gaining ground. Cities such as Salt Lake City, Denver, and Raleigh-Durham show rapid AI job growth relative to population, aided by remote hiring and migration of skilled workers. This suggests that distributed innovation ecosystems are replacing the singular dominance of coastal tech centers.



Top 15 Tech Hubs

## Urban vs. Rural Dynamics

Urban regions continue to host the majority of AI employment due to their access to universities, data infrastructure, and venture capital. However, remote work options are beginning to bridge the urban-rural divide, allowing rural professionals to participate in digital industries without relocating. Rural job markets, while smaller, show higher proportions of non-AI and on-site work, reflecting continued dependence on physical industries and public sector employment.



Urban and Rural AI Makeup

### References

Khan, A., Shad, F., Sethi, S., & Bibi, M. (2024). The impact of artificial intelligence (AI) on job displacement and the future of work. Social Science Review Archives, 2(2), 2296–2306. https://doi.org/10.70670/SRA.v3i1.509

Sheffi, Y. (2024). Technology is not enough: Potential job displacement in an AI-driven future. Journal of Supply Chain Management, Logistics and Procurement, 6(4), 338–351.

Solo, L. B., Hossain, S., & Weah, S. S. II. (2025). AI-powered job market insights: How AI adoption influences salary and job growth projections. North American Academic Research, 8(2), 224–241. https://doi.org/10.5281/zenodo.15212191

# Load and Preview LightCast Data

from pyspark.sql import SparkSession  
  
  
# Start a Spark session  
spark = SparkSession.builder.appName("JobPostingsAnalysis").getOrCreate()  
  
# Load the CSV file into a Spark DataFrame  
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","true").option("escape", "\"").csv("../data/lightcast\_job\_postings.csv")

WARNING: Using incubator modules: jdk.incubator.vector  
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
25/10/16 02:35:14 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
[Stage 1:> (0 + 1) / 1]

# Drop unnecessary columns

Redundant or irrelevant columns are dropped here.

NAICS and SOC levels are removed because each job is already described by its most detailed (and most general) industry and occupation classification (NAICS\_2022\_6\_NAME and SOC\_6\_NAME).

Timestamps and system variables (like LAST\_UPDATED\_TIMESTAMP) are not meaningful.

Simplifying the dataset this way speeds up our processing and makes the whole dataframe look more clean and user-friendly.

import pandas as pd  
from pyspark.sql.functions import when, col  
  
#Clean Data to convert to Pandas  
columns\_to\_drop = ["ID", "BODY", "URL", "ACTIVE\_URLS", "DUPLICATES", "LAST\_UPDATED\_TIMESTAMP",  
 "NAICS2", "NAICS3", "NAICS4", "NAICS5", "NAICS6",  
 "SOC\_2", "SOC\_3", "SOC\_4", "SOC\_5", "LAST\_UPDATED\_DATE", "LAST\_UPDATED\_TIMESTAMP", "EXPIRED", "SOURCE\_TYPES", "SOURCES", "ACTIVE\_SOURCES\_INFO", "MODELED\_EXPIRED", "MODELED\_DURATION", "NAICS2\_NAME", "NAICS3\_NAME", "NAICS4\_NAME", "NAICS5\_NAME", "NAICS6\_NAME",  
 "SOC\_2\_NAME", "SOC\_3\_NAME", "SOC\_4\_NAME", "SOC\_5\_NAME", "EDUCATION\_LEVELS", "MIN\_EDULEVELS"  
   
]  
cleaned\_data = df.drop(\*columns\_to\_drop)  
  
cleaned\_data = cleaned\_data.withColumn(  
 "REMOTE\_TYPE\_NAME",  
 when(col("REMOTE\_TYPE\_NAME") == "Remote", "Remote")  
 .when(col("REMOTE\_TYPE\_NAME") == "Hybrid Remote", "Hybrid")  
 .when(col("REMOTE\_TYPE\_NAME") == "[None]", "On-site")  
 .when(col("REMOTE\_TYPE\_NAME").isNull(), "On-site")  
 .when(col("REMOTE\_TYPE\_NAME") == "Not Remote", "On-site")  
 .otherwise(col("REMOTE\_TYPE\_NAME"))  
)  
  
cleaned\_data = cleaned\_data.withColumn(  
 "EMPLOYMENT\_TYPE\_NAME",  
 when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time / full-time", "Flexible")  
 .when(col("EMPLOYMENT\_TYPE\_NAME").isNull(), "Full-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time (â‰¤ 32 hours)", "Part-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Full-time (> 32 hours)", "Full-Time")  
 .otherwise(col("EMPLOYMENT\_TYPE\_NAME"))   
)  
  
cleaned\_data = cleaned\_data.filter(col("NAICS\_2022\_2\_NAME") != "Unclassified Industry")  
  
median\_salary = cleaned\_data.approxQuantile("SALARY", [0.5], 0.01)[0]  
cleaned\_data = cleaned\_data.withColumn(  
 "SALARY",  
 when(col("SALARY").isNull(), median\_salary).otherwise(col("SALARY"))  
)  
  
  
clean\_pdf = cleaned\_data.toPandas()

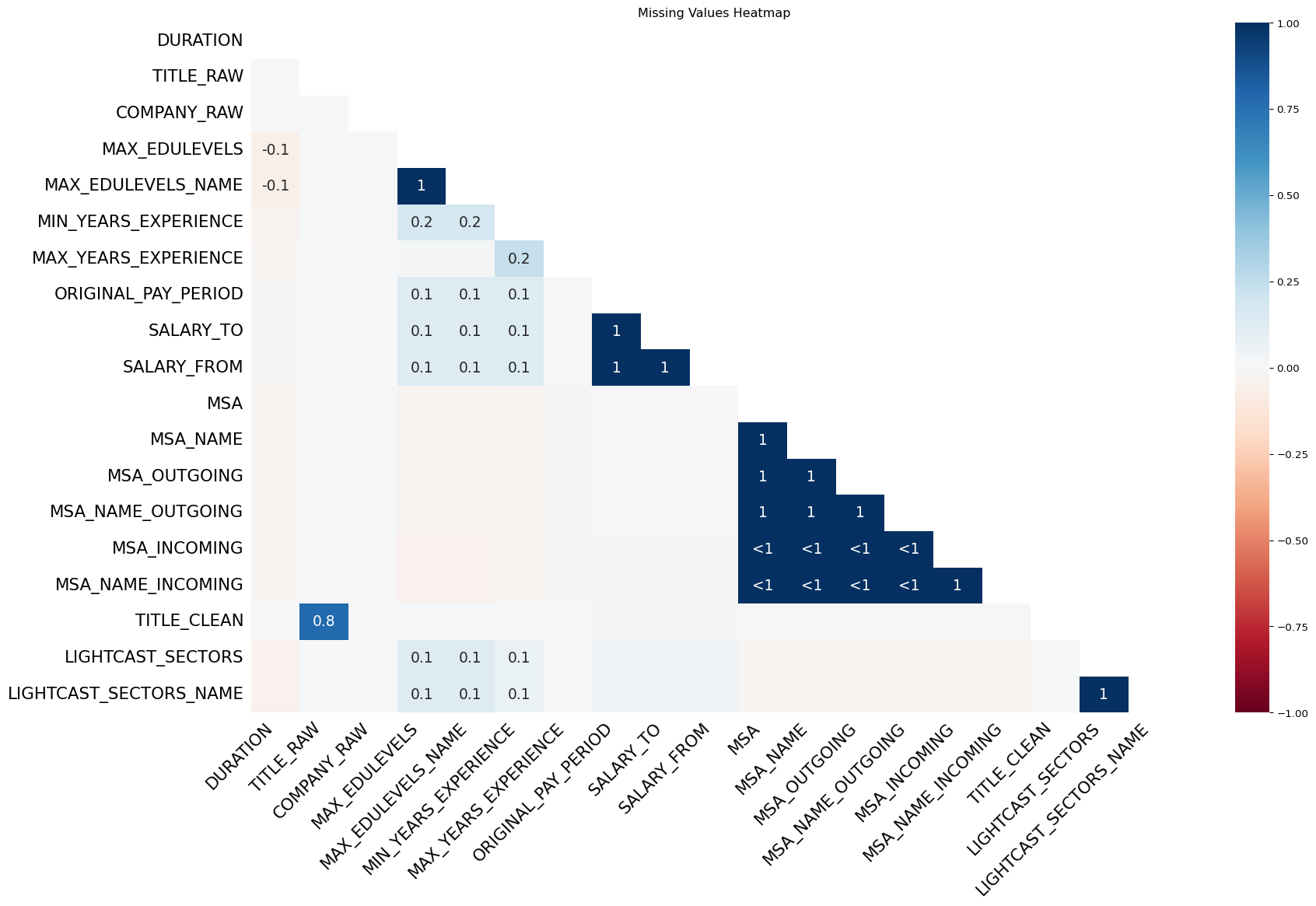
[Stage 2:> (0 + 1) / 1] 25/10/16 02:35:34 WARN SparkStringUtils: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.sql.debug.maxToStringFields'.  
[Stage 3:> (0 + 1) / 1]

# Handle Missing Values

We also cleaned categorical values

Missing categorical data (like City, Company, State) were replaced with “Unknown” so that there is data in all rows. Duplicates were also dropped to not skew the analysis. Salary missing values were replaced with the median salary. Remote Type Name and Employment type were simplified into smaller groupings.

import missingno as msno  
import matplotlib.pyplot as plt  
  
# Visualize missing data  
msno.heatmap(clean\_pdf)  
plt.title("Missing Values Heatmap")  
plt.show()  
  
fill\_cols = ["CITY\_NAME", "CITY", "LOCATION", "STATE", "STATE\_NAME", "COMPANY", "COMPANY\_NAME"]  
clean\_pdf[fill\_cols] = clean\_pdf[fill\_cols].fillna("Unknown")  
  
clean\_pdf = clean\_pdf.drop\_duplicates(subset=["TITLE", "COMPANY", "LOCATION", "POSTED"], keep="first")  
  
clean\_pdf.dropna(thresh=len(clean\_pdf)\*0.5, axis=1, inplace=True)



# Helper Columns for classifying AI and Posted Dates

We created columns to classify what job titles may be affected by AI vs Non-AI Job Titles We created a column for the month that the job was posted in order to create growth data

#New Column to Classify AI Jobs and Add Month of Posting Date  
  
  
ai\_keywords = [  
 "AI", "Machine Learning", "Data Scientist", "Data Analyst", "ML",   
 "Artificial Intelligence", "Deep Learning", "NLP", "Predictive Analytics"  
]  
  
#Function to classify AI vs Non-AI Jobs  
def classify\_ai(title):  
 title\_lower = str(title).lower()  
 for keyword in ai\_keywords:  
 if keyword.lower() in title\_lower:  
 return "AI"  
 return "Non-AI"  
  
clean\_pdf["AI\_JOB"] = clean\_pdf["TITLE\_RAW"].apply(classify\_ai)  
  
clean\_pdf["POSTED"] = pd.to\_datetime(clean\_pdf["POSTED"], errors="coerce")  
clean\_pdf["POSTED\_MONTH"] = clean\_pdf["POSTED"].dt.month  
  
clean\_pdf.head(25)

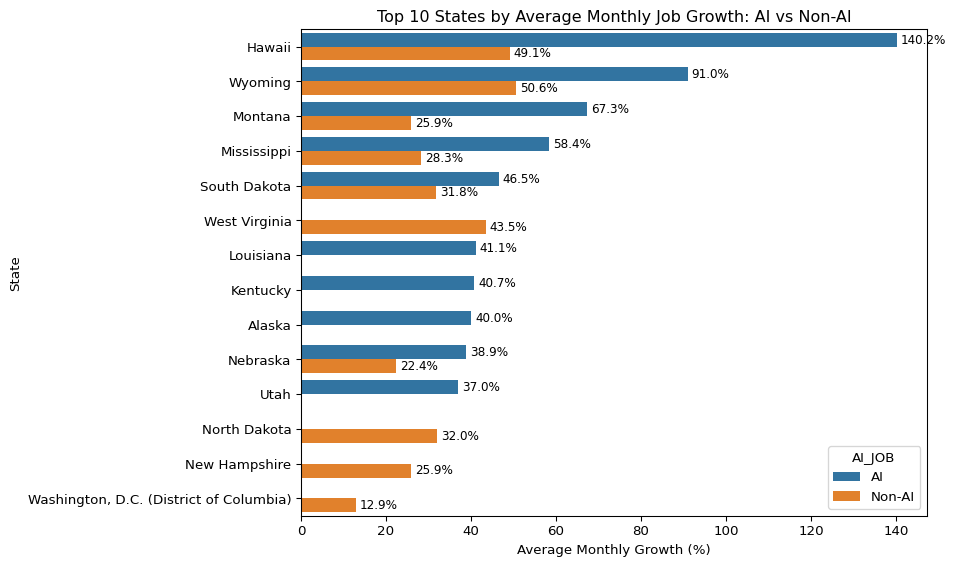
|  | POSTED | DURATION | TITLE\_RAW | COMPANY | COMPANY\_NAME | COMPANY\_RAW | COMPANY\_IS\_STAFFING | EDUCATION\_LEVELS\_NAME | MIN\_EDULEVELS\_NAME | EMPLOYMENT\_TYPE | ... | NAICS\_2022\_3 | NAICS\_2022\_3\_NAME | NAICS\_2022\_4 | NAICS\_2022\_4\_NAME | NAICS\_2022\_5 | NAICS\_2022\_5\_NAME | NAICS\_2022\_6 | NAICS\_2022\_6\_NAME | AI\_JOB | POSTED\_MONTH |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 2024-06-02 | 6.0 | Enterprise Analyst (II-III) | 894731 | Murphy USA | Murphy USA | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 441 | Motor Vehicle and Parts Dealers | 4413 | Automotive Parts, Accessories, and Tire Retailers | 44133 | Automotive Parts and Accessories Retailers | 441330 | Automotive Parts and Accessories Retailers | Non-AI | 6 |
| 1 | 2024-06-02 | NaN | Oracle Consultant - Reports (3592) | 133098 | Smx Corporation Limited | SMX | True | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 561 | Administrative and Support Services | 5613 | Employment Services | 56132 | Temporary Help Services | 561320 | Temporary Help Services | Non-AI | 6 |
| 2 | 2024-06-02 | 35.0 | Data Analyst | 39063746 | Sedgwick | Sedgwick | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 524 | Insurance Carriers and Related Activities | 5242 | Agencies, Brokerages, and Other Insurance Rela... | 52429 | Other Insurance Related Activities | 524291 | Claims Adjusting | AI | 6 |
| 3 | 2024-06-02 | 48.0 | Sr. Lead Data Mgmt. Analyst - SAS Product Owner | 37615159 | Wells Fargo | Wells Fargo | False | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 522 | Credit Intermediation and Related Activities | 5221 | Depository Credit Intermediation | 52211 | Commercial Banking | 522110 | Commercial Banking | Non-AI | 6 |
| 4 | 2024-06-02 | 10.0 | SR Lead Data Analyst | 2233642 | Lumen Technologies | Lumen | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 517 | Telecommunications | 5178 | All Other Telecommunications | 51781 | All Other Telecommunications | 517810 | All Other Telecommunications | AI | 6 |
| 5 | 2024-06-02 | NaN | Talent Data Analyst | 44896740 | Semiconductor Components Industries | Semiconductor Components Industries, LLC | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 334 | Computer and Electronic Product Manufacturing | 3344 | Semiconductor and Other Electronic Component M... | 33441 | Semiconductor and Other Electronic Component M... | 334413 | Semiconductor and Related Device Manufacturing | AI | 6 |
| 6 | 2024-06-02 | 35.0 | Data Analyst | 39063746 | Sedgwick | Sedgwick | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 524 | Insurance Carriers and Related Activities | 5242 | Agencies, Brokerages, and Other Insurance Rela... | 52429 | Other Insurance Related Activities | 524291 | Claims Adjusting | AI | 6 |
| 7 | 2024-06-02 | NaN | Sr. Marketing Analyst | 39016169 | Dassault SystÃ¨mes | Dassault Systmes | False | [\n "Bachelor's degree",\n "Master's degree"\n] | Bachelor's degree | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5415 | Computer Systems Design and Related Services | 54151 | Computer Systems Design and Related Services | 541511 | Custom Computer Programming Services | Non-AI | 6 |
| 8 | 2024-06-02 | NaN | Data Analyst | 12147696 | DCS Corporation | DCS Corp. | False | [\n "High school or GED",\n "Associate degre... | High school or GED | 1 | ... | 423 | Merchant Wholesalers, Durable Goods | 4238 | Machinery, Equipment, and Supplies Merchant Wh... | 42383 | Industrial Machinery and Equipment Merchant Wh... | 423830 | Industrial Machinery and Equipment Merchant Wh... | AI | 6 |
| 9 | 2024-06-02 | 6.0 | Data Analyst | 4063994 | Allegis Group | TEKsystems c/o Allegis Group | True | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 561 | Administrative and Support Services | 5613 | Employment Services | 56132 | Temporary Help Services | 561320 | Temporary Help Services | AI | 6 |
| 10 | 2024-06-02 | 33.0 | Data Research Analyst - J00156981 | 34294036 | Equifax | Equifax, Inc. | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 522 | Credit Intermediation and Related Activities | 5223 | Activities Related to Credit Intermediation | 52232 | Financial Transactions Processing, Reserve, an... | 522320 | Financial Transactions Processing, Reserve, an... | Non-AI | 6 |
| 11 | 2024-06-02 | 55.0 | Power, Utilities & Renewables Consulting Manag... | 5732448 | Deloitte | Deloitte | False | [\n "Bachelor's degree",\n "Master's degree"\n] | Bachelor's degree | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5416 | Management, Scientific, and Technical Consulti... | 54161 | Management Consulting Services | 541611 | Administrative Management and General Manageme... | Non-AI | 6 |
| 12 | 2024-06-02 | NaN | Sr. Enterprise Data Architecture | 38205299 | Lincoln Financial Group | Lincoln Financial Group | False | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 523 | Securities, Commodity Contracts, and Other Fin... | 5239 | Other Financial Investment Activities | 52394 | Portfolio Management and Investment Advice | 523940 | Portfolio Management and Investment Advice | Non-AI | 6 |
| 13 | 2024-06-02 | NaN | SENIOR CONSULTANT, Continuous Improvement & Da... | 1967 | Boston University | Boston University | False | [\n "Associate degree",\n "Bachelor's degree... | Associate degree | 1 | ... | 611 | Educational Services | 6113 | Colleges, Universities, and Professional Schools | 61131 | Colleges, Universities, and Professional Schools | 611310 | Colleges, Universities, and Professional Schools | Non-AI | 6 |
| 14 | 2024-06-02 | 18.0 | SAP FSCM Consultant | 8592955 | Accenture | Accenture | False | [\n "Associate degree",\n "Bachelor's degree... | Associate degree | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5415 | Computer Systems Design and Related Services | 54151 | Computer Systems Design and Related Services | 541512 | Computer Systems Design Services | Non-AI | 6 |
| 15 | 2024-06-02 | NaN | Oracle Consultant - Reports (3592) | 133098 | Smx Corporation Limited | SMX | True | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 561 | Administrative and Support Services | 5613 | Employment Services | 56132 | Temporary Help Services | 561320 | Temporary Help Services | Non-AI | 6 |
| 16 | 2024-06-02 | NaN | Principal Architect | 39192167 | Surgical Care Affiliates | Surgical Care Affiliates SCA | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 621 | Ambulatory Health Care Services | 6214 | Outpatient Care Centers | 62149 | Other Outpatient Care Centers | 621493 | Freestanding Ambulatory Surgical and Emergency... | Non-AI | 6 |
| 18 | 2024-06-02 | 46.0 | Senior Enterprise Architect (remote virtual) | 6915630 | Humana | Humana | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 524 | Insurance Carriers and Related Activities | 5241 | Insurance Carriers | 52411 | Direct Life, Health, and Medical Insurance Car... | 524114 | Direct Health and Medical Insurance Carriers | Non-AI | 6 |
| 19 | 2024-06-02 | 20.0 | DATA ANALYTICS | 99484525 | BCforward | BCforward | True | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5416 | Management, Scientific, and Technical Consulti... | 54161 | Management Consulting Services | 541611 | Administrative Management and General Manageme... | Non-AI | 6 |
| 20 | 2024-06-02 | NaN | Enterprise Architect | 44726222 | Red Cedar Consultancy | Red Cedar Consultancy LLC | False | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5416 | Management, Scientific, and Technical Consulti... | 54161 | Management Consulting Services | 541618 | Other Management Consulting Services | Non-AI | 6 |
| 21 | 2024-06-02 | NaN | Process Engineer | 3995324 | Michelin | Michelin | False | [\n "Bachelor's degree"\n] | Bachelor's degree | 1 | ... | 326 | Plastics and Rubber Products Manufacturing | 3262 | Rubber Product Manufacturing | 32621 | Tire Manufacturing | 326211 | Tire Manufacturing (except Retreading) | Non-AI | 6 |
| 22 | 2024-06-02 | 16.0 | Data Analyst Level 4 | 99487184 | Ic-Cap | IC-CAP, LLC | False | [\n "High school or GED",\n "Bachelor's degr... | High school or GED | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5416 | Management, Scientific, and Technical Consulti... | 54169 | Other Scientific and Technical Consulting Serv... | 541690 | Other Scientific and Technical Consulting Serv... | AI | 6 |
| 23 | 2024-06-02 | NaN | Security Enterprise Architect | 41994151 | Genesis10 | GENESIS 10 | True | [\n "No Education Listed"\n] | No Education Listed | 1 | ... | 541 | Professional, Scientific, and Technical Services | 5415 | Computer Systems Design and Related Services | 54151 | Computer Systems Design and Related Services | 541512 | Computer Systems Design Services | Non-AI | 6 |
| 24 | 2024-06-02 | 12.0 | Client Quantitative Analyst I | 4808322 | Bank of America | Bank of America | False | [\n "Bachelor's degree",\n "Master's degree"\n] | Bachelor's degree | 1 | ... | 522 | Credit Intermediation and Related Activities | 5223 | Activities Related to Credit Intermediation | 52232 | Financial Transactions Processing, Reserve, an... | 522320 | Financial Transactions Processing, Reserve, an... | Non-AI | 6 |
| 25 | 2024-06-02 | NaN | Senior Analyst - Data Analytics and Reporting | 897843 | Carlson | Carlson | False | [\n "Bachelor's degree",\n "Master's degree"\n] | Bachelor's degree | 1 | ... | 812 | Personal and Laundry Services | 8121 | Personal Care Services | 81211 | Hair, Nail, and Skin Care Services | 812112 | Beauty Salons | Non-AI | 6 |

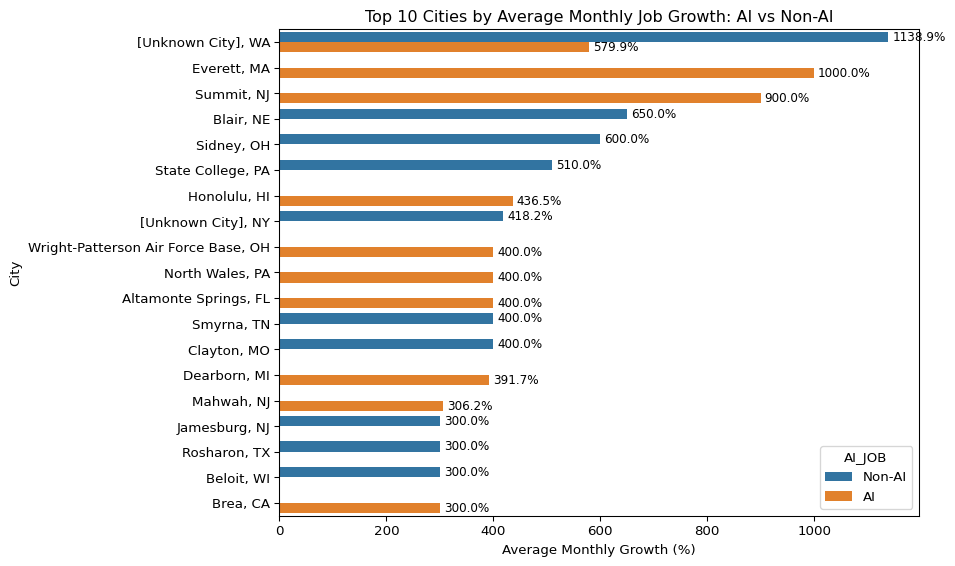
# City and State Analysis by AI vs Non-AI Jobs

Key Insights:

#Question 1 Visualization: Which Cities or States have the highest job growth for AI vs Non-AI  
  
count\_by\_month\_state = (  
 clean\_pdf.groupby(["STATE\_NAME", "POSTED\_MONTH", "AI\_JOB"])  
 .size()  
 .reset\_index(name="count")  
)  
  
count\_by\_month\_city = (  
 clean\_pdf.groupby(["CITY\_NAME", "POSTED\_MONTH", "AI\_JOB"])  
 .size()  
 .reset\_index(name="count")  
)  
  
#Measure job growth by State and then by city  
count\_by\_month\_state = count\_by\_month\_state.sort\_values(["STATE\_NAME", "AI\_JOB", "POSTED\_MONTH"])  
count\_by\_month\_state["GROWTH"] = (  
 count\_by\_month\_state  
 .groupby(["STATE\_NAME", "AI\_JOB"])["count"]  
 .pct\_change() \* 100  
)  
  
count\_by\_month\_city = count\_by\_month\_city.sort\_values(["CITY\_NAME", "AI\_JOB", "POSTED\_MONTH"])  
count\_by\_month\_city["GROWTH"] = (  
 count\_by\_month\_city  
 .groupby(["CITY\_NAME", "AI\_JOB"])["count"]  
 .pct\_change() \* 100  
)  
  
avg\_growth\_state = (  
 count\_by\_month\_state.groupby(["STATE\_NAME", "AI\_JOB"])["GROWTH"]  
 .mean()  
 .reset\_index()  
 .dropna()  
 .sort\_values("GROWTH", ascending=False)  
)  
  
avg\_growth\_city = (  
 count\_by\_month\_city.groupby(["CITY\_NAME", "AI\_JOB"])["GROWTH"]  
 .mean()  
 .reset\_index()  
 .dropna()  
 .sort\_values("GROWTH", ascending=False)  
)  
  
print("Top 10 States by AI Job Growth:")  
print(avg\_growth\_state[avg\_growth\_state["AI\_JOB"] == "AI"].head(10))  
  
print("\nTop 10 States by Non-AI Job Growth:")  
print(avg\_growth\_state[avg\_growth\_state["AI\_JOB"] == "Non-AI"].head(10))  
  
print("Top 10 Cities by AI Job Growth:")  
print(avg\_growth\_city[avg\_growth\_city["AI\_JOB"] == "AI"].head(10))  
  
print("\nTop 10 Cities by Non-AI Job Growth:")  
print(avg\_growth\_city[avg\_growth\_city["AI\_JOB"] == "Non-AI"].head(10))  
  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
state\_visual = avg\_growth\_state.groupby("AI\_JOB").head(10)  
  
plt.figure(figsize=(10,6))  
ax\_state = sns.barplot(data=state\_visual, y="STATE\_NAME", x="GROWTH", hue="AI\_JOB")  
plt.title("Top 10 States by Average Monthly Job Growth: AI vs Non-AI")  
plt.xlabel("Average Monthly Growth (%)")  
plt.ylabel("State")  
  
for container in ax\_state.containers:  
 ax\_state.bar\_label(container, fmt="%.1f%%", label\_type="edge", padding=3, fontsize=9)  
  
plt.tight\_layout()  
plt.savefig("../ad688\_group6\_geographic\_analysis/images/top10state.png", dpi=300)  
plt.show()  
  
city\_visual = avg\_growth\_city.groupby("AI\_JOB").head(10)  
  
plt.figure(figsize=(10,6))  
ax\_city = sns.barplot(data=city\_visual, y="CITY\_NAME", x="GROWTH", hue="AI\_JOB")  
plt.title("Top 10 Cities by Average Monthly Job Growth: AI vs Non-AI")  
plt.xlabel("Average Monthly Growth (%)")  
plt.ylabel("City")  
  
for container in ax\_city.containers:  
 ax\_city.bar\_label(container, fmt="%.1f%%", label\_type="edge", padding=3, fontsize=9)  
  
plt.tight\_layout()  
plt.savefig("../ad688\_group6\_geographic\_analysis/images/top10city.png", dpi=300)  
plt.show()

Top 10 States by AI Job Growth:  
 STATE\_NAME AI\_JOB GROWTH  
20 Hawaii AI 140.217803  
100 Wyoming AI 90.990260  
50 Montana AI 67.291667  
46 Mississippi AI 58.393822  
80 South Dakota AI 46.527778  
34 Louisiana AI 41.077075  
32 Kentucky AI 40.737045  
2 Alaska AI 40.043290  
52 Nebraska AI 38.870132  
86 Utah AI 36.988636  
  
Top 10 States by Non-AI Job Growth:  
 STATE\_NAME AI\_JOB GROWTH  
101 Wyoming Non-AI 50.555556  
21 Hawaii Non-AI 49.105634  
97 West Virginia Non-AI 43.492753  
67 North Dakota Non-AI 32.006313  
81 South Dakota Non-AI 31.818182  
47 Mississippi Non-AI 28.331625  
57 New Hampshire Non-AI 25.913029  
51 Montana Non-AI 25.904481  
53 Nebraska Non-AI 22.365222  
95 Washington, D.C. (District of Columbia) Non-AI 12.850958  
Top 10 Cities by AI Job Growth:  
 CITY\_NAME AI\_JOB GROWTH  
1366 Everett, MA AI 1000.000000  
4155 Summit, NJ AI 900.000000  
4876 [Unknown City], WA AI 579.875000  
1964 Honolulu, HI AI 436.507937  
4743 Wright-Patterson Air Force Base, OH AI 400.000000  
3133 North Wales, PA AI 400.000000  
82 Altamonte Springs, FL AI 400.000000  
1072 Dearborn, MI AI 391.666667  
2532 Mahwah, NJ AI 306.250000  
474 Brea, CA AI 300.000000  
  
Top 10 Cities by Non-AI Job Growth:  
 CITY\_NAME AI\_JOB GROWTH  
4877 [Unknown City], WA Non-AI 1138.925972  
396 Blair, NE Non-AI 650.000000  
3974 Sidney, OH Non-AI 600.000000  
4106 State College, PA Non-AI 510.000000  
4851 [Unknown City], NY Non-AI 418.161765  
4003 Smyrna, TN Non-AI 400.000000  
822 Clayton, MO Non-AI 400.000000  
2101 Jamesburg, NJ Non-AI 300.000000  
3720 Rosharon, TX Non-AI 300.000000  
323 Beloit, WI Non-AI 300.000000

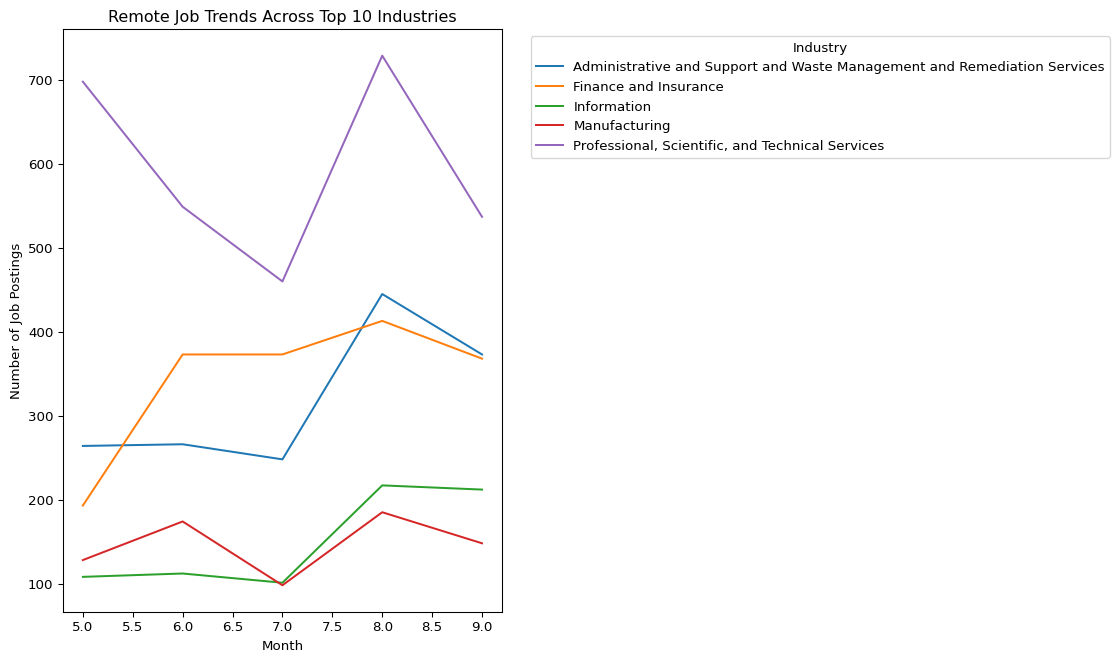




# Remote Job Growth by Industry

Key Insights:

#Question 2: Are remote jobs increasing or decreasing across industries?  
  
remote\_only = clean\_pdf[clean\_pdf["REMOTE\_TYPE\_NAME"] == "Remote"]  
  
remote\_growth = (  
 remote\_only.groupby(["NAICS\_2022\_2\_NAME", "POSTED\_MONTH"])  
 .size()  
 .reset\_index(name="count")  
)  
  
  
top\_5\_industries = (  
 remote\_only["NAICS\_2022\_2\_NAME"]  
 .value\_counts()  
 .head(5)  
 .index  
)  
  
top\_remote\_growth = remote\_growth[remote\_growth["NAICS\_2022\_2\_NAME"].isin(top\_5\_industries)]  
  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(12, 7))  
sns.lineplot(  
 data=top\_remote\_growth,  
 x="POSTED\_MONTH",  
 y="count",  
 hue="NAICS\_2022\_2\_NAME"  
)  
plt.title("Remote Job Trends Across Top 10 Industries")  
plt.xlabel("Month")  
plt.ylabel("Number of Job Postings")  
plt.legend(title="Industry", bbox\_to\_anchor=(1.05, 1), loc='upper left')  
plt.tight\_layout()  
plt.savefig("../ad688\_group6\_geographic\_analysis/images/remoteindustries.png", dpi=300)  
plt.show()



# Tech-Hubs vs emerging Markets

Key Insights:

#Question #3: Do Tech hubs (Silicon Valley, Austin, Boston) still dominate hiring, or are other locations emerging?  
  
tech\_hubs = ["Austin-Round Rock-Georgetown, TX", "Boston-Cambridge-Newton, MA-NH","Los Angeles-Long Beach-Anaheim, CA","San Diego-Chula Vista-Carlsbad, CA","San Francisco-Oakland-Berkeley, CA","San Jose-Sunnyvale-Santa Clara, CA","Seattle-Tacoma-Bellevue, WA"]  
  
# Create a column classifying if the city is a tech hub  
clean\_pdf["TECH\_HUB"] = clean\_pdf["MSA\_NAME"].apply(  
 lambda x: "Tech Hub" if x in tech\_hubs else "Other"  
)  
  
#Count number of postings by Tech Hub  
tech\_hub\_counts = (  
 clean\_pdf.groupby("MSA\_NAME")  
 .size()  
 .reset\_index(name="count")  
 .sort\_values("count", ascending=False)  
)  
  
# Merge to add TECH\_HUB classification to each MSA  
tech\_hub\_counts = tech\_hub\_counts.merge(  
 clean\_pdf[["MSA\_NAME", "TECH\_HUB"]].drop\_duplicates(),  
 on="MSA\_NAME",  
 how="left"  
)  
  
  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
plt.figure(figsize=(12, 8))  
sns.barplot(  
 data=tech\_hub\_counts.head(15),  
 x="MSA\_NAME",  
 y="count",  
 hue="TECH\_HUB",  
 palette="viridis"  
)  
plt.title("Top 15 Metro Areas by Count of Job Postings")  
plt.xlabel("Tech Hub")  
plt.ylabel("Number of Job Postings")  
plt.xticks(rotation=60, ha="right")  
  
# Add labels on top of bars  
for i, row in tech\_hub\_counts.head(15).iterrows():  
 plt.text(row["count"] + 100, i, f"{row['count']:,}", va="center", fontsize=10)  
  
plt.tight\_layout()  
plt.savefig("../ad688\_group6\_geographic\_analysis/images/top15techhubs.png", dpi=300)  
plt.show()

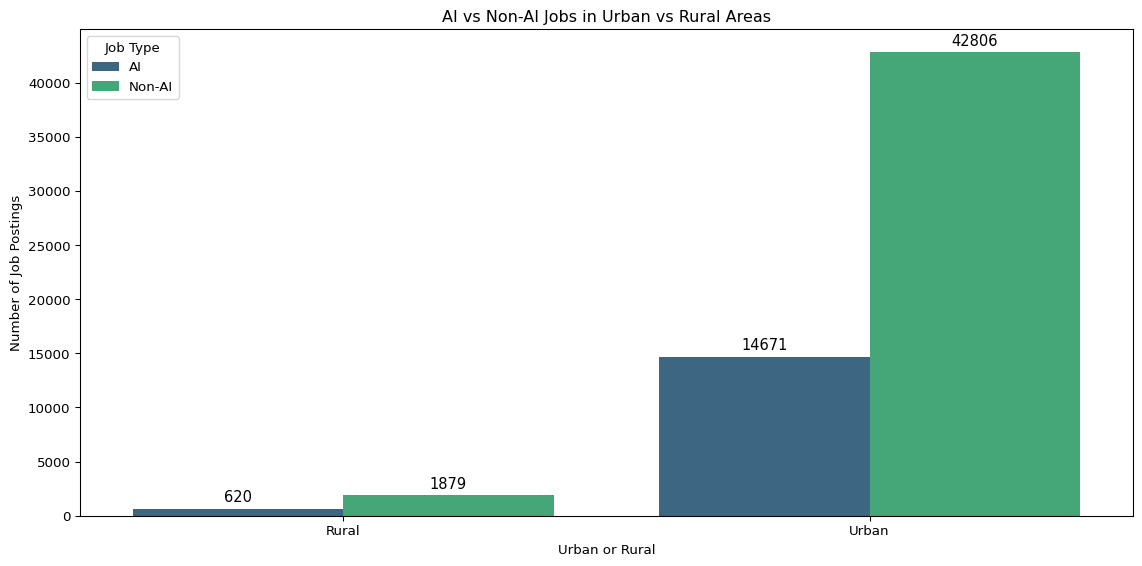
/tmp/ipykernel\_7310/218927844.py:46: UserWarning:  
  
Tight layout not applied. The left and right margins cannot be made large enough to accommodate all Axes decorations.

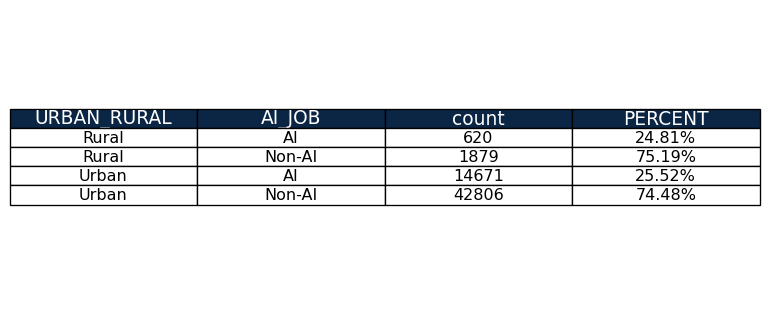


# A Comparison of the Urban and Rural Job Market in relation to AI Careers

Key Insights:

# Question 4: How do Urban vs. Rural Job markets differ for AI and non-AI careers?  
  
# Classify as 'Urban' if MSA\_NAME is present, else 'Rural'  
clean\_pdf["URBAN\_RURAL"] = clean\_pdf["MSA\_NAME"].apply(lambda x: "Urban" if pd.notnull(x) else "Rural")  
  
# Group data by month, urban/rural, and AI vs Non-AI  
urban\_rural\_jobs = (  
 clean\_pdf.groupby(["URBAN\_RURAL", "AI\_JOB"])  
 .size()  
 .reset\_index(name="count")  
)  
  
#Calculate percentages  
urban\_rural\_jobs["PERCENT"] = (  
 urban\_rural\_jobs.groupby("URBAN\_RURAL")["count"]  
 .apply(lambda x: 100 \* x / x.sum())  
 .values  
)  
  
#Convert Percentage into 2 decimal places  
  
urban\_rural\_jobs["PERCENT"] = urban\_rural\_jobs["PERCENT"].apply(lambda x:f"{x:.2f}%")  
  
# Visualization  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(12, 6))  
ax\_urban = sns.barplot(  
 data=urban\_rural\_jobs,  
 x="URBAN\_RURAL",  
 y="count",  
 hue="AI\_JOB",  
 palette="viridis"  
)  
  
plt.title("AI vs Non-AI Jobs in Urban vs Rural Areas")  
plt.xlabel("Urban or Rural")  
plt.ylabel("Number of Job Postings")  
plt.legend(title="Job Type")  
  
for container in ax\_urban.containers:  
 ax\_urban.bar\_label(container, fmt="%d", label\_type="edge", padding=3, fontsize=11)  
plt.tight\_layout()  
plt.savefig("../ad688\_group6\_geographic\_analysis/images/urbanai.png", dpi=300)  
plt.show()  
  
fig, ax = plt.subplots(figsize=(7,4))   
ax.axis('off')   
  
table = ax.table(  
 cellText=urban\_rural\_jobs.values,  
 colLabels=urban\_rural\_jobs.columns,  
 cellLoc='center',  
 loc='center',  
 colColours=["#0b2545"]\*len(urban\_rural\_jobs.columns), # Dark blue header  
 colWidths=[0.3]\*len(urban\_rural\_jobs.columns)  
)  
  
table.auto\_set\_font\_size(False)  
table.set\_fontsize(12)  
table.scale(1.2, 1.2)   
  
for key, cell in table.get\_celld().items():  
 if key[0] == 0:  
 cell.set\_fontsize(14)  
 cell.set\_text\_props(color='white')  
 cell.set\_facecolor('#0b2545')  
  
plt.show()





## title: “Skills Gap Analysis”

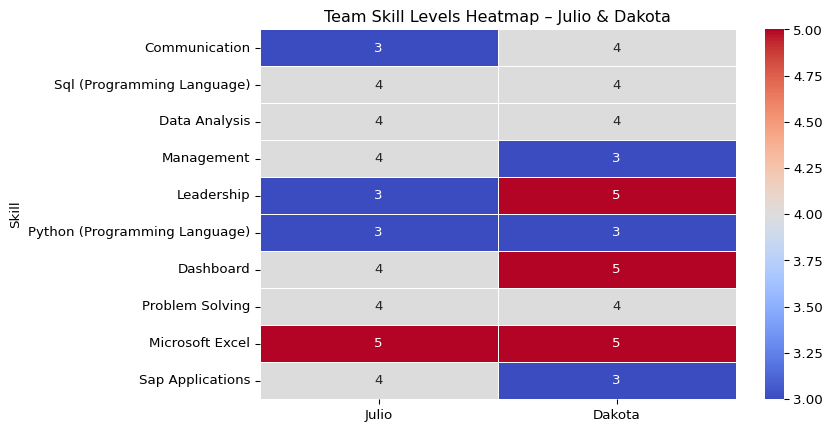
# Skills Gap Analysis

from pyspark.sql import SparkSession  
  
  
# Start a Spark session  
spark = SparkSession.builder.appName("JobPostingsAnalysis").getOrCreate()  
  
# Load the CSV file into a Spark DataFrame  
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","true").option("escape", "\"").csv("../data/lightcast\_job\_postings.csv")

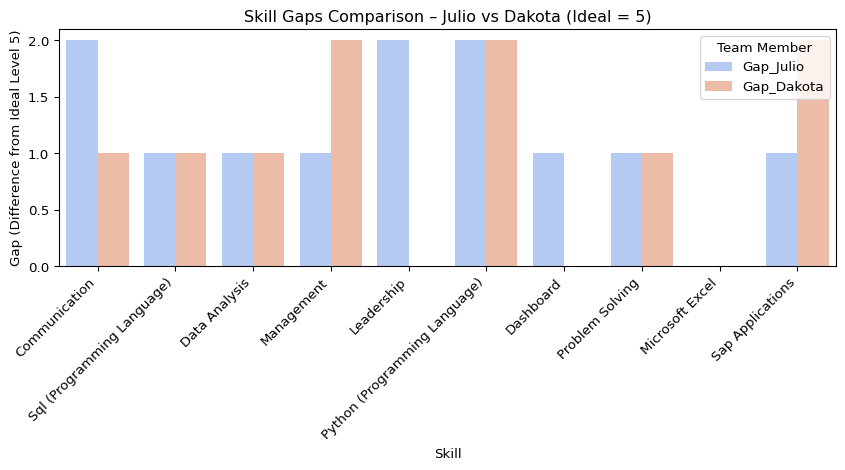
[Stage 5:> (0 + 1) / 1]

import pandas as pd  
  
columns = [  
 "COMPANY\_NAME", "COMPANY\_IS\_STAFFING", # Identification, company  
 "POSTED", "EXPIRED", "DURATION", "MODELED\_DURATION", # Dates, duration  
 "TITLE\_NAME", "EMPLOYMENT\_TYPE\_NAME", "IS\_INTERNSHIP", # Job title, contract type  
 "CITY\_NAME", "STATE\_NAME", "REMOTE\_TYPE\_NAME", # Geographic  
 "MIN\_YEARS\_EXPERIENCE", "MIN\_EDULEVELS\_NAME", "EDUCATION\_LEVELS\_NAME", # Education, experience  
 "SALARY", # Salary  
 "SKILLS\_NAME", "SPECIALIZED\_SKILLS\_NAME", "SOFTWARE\_SKILLS\_NAME", # Tech skills  
 "COMMON\_SKILLS\_NAME", # Common, soft skills  
 "CERTIFICATIONS\_NAME" # Certif  
]  
  
  
df\_columns = df.limit(4000).select([c for c in columns if c in df.columns])  
df\_columns\_pd=df\_columns.toPandas()  
  
skill\_cols = [  
"SKILLS\_NAME",  
"SPECIALIZED\_SKILLS\_NAME",  
"SOFTWARE\_SKILLS\_NAME",  
"COMMON\_SKILLS\_NAME",  
"CERTIFICATIONS\_NAME"  
]  
  
df\_columns\_pd["ALL\_SKILLS\_RAW"] = df\_columns\_pd[skill\_cols].fillna("").agg(" ".join, axis=1)  
df\_columns\_pd["ALL\_SKILLS\_RAW"] = (  
 df\_columns\_pd[skill\_cols]  
 .fillna("")  
 .agg(" ".join, axis=1)  
 .astype(str)  
 .str.replace(r"[{}\[\]'\"]", "", regex=True) # quita corchetes y comillas  
 .str.replace(r"\b[Nn]one\b|nan", "", regex=True) # quita None/nan  
 .str.replace(r"[;|/]", ",", regex=True) # normaliza separadores  
 .str.replace(r"\s\*,\s\*", ", ", regex=True) # limpia espacios entre comas  
 .str.replace(r"\s{2,}", " ", regex=True) # elimina espacios dobles  
 .str.strip() # quita espacios extra  
)  
print(df\_columns\_pd["ALL\_SKILLS\_RAW"].head(5))  
pd.set\_option('display.max\_colwidth', None); print(df\_columns\_pd["ALL\_SKILLS\_RAW"].head(5).to\_string(index=False))  
  
  
  
# skills text to list.  
df\_columns\_pd["ALL\_SKILLS\_LIST"] = df\_columns\_pd["ALL\_SKILLS\_RAW"].str.split(",")  
  
# list to row  
df\_skills = df\_columns\_pd.explode("ALL\_SKILLS\_LIST")  
  
# Clean up spaces and drop empty rows  
df\_skills = ( df\_skills.dropna(subset=["ALL\_SKILLS\_LIST"]).loc[df\_skills["ALL\_SKILLS\_LIST"].str.strip() != ""])  
df\_skills["ALL\_SKILLS\_LIST"] = ( df\_skills["ALL\_SKILLS\_LIST"] .str.strip() .str.title())  
  
# Count skills  
top\_skills = (df\_skills["ALL\_SKILLS\_LIST"].value\_counts().reset\_index().rename(columns={"index": "Skill", "ALL\_SKILLS\_LIST": "Frequency"}))  
  
# Show top 20  
print(top\_skills.head(20))  
  
# Frequency count  
# 0 Communication 3394  
# 1 Sql (Programming Language) 3134  
# 2 Data Analysis 2960  
# 3 Management 2116  
# 4 Leadership 2023  
# 5 Python (Programming Language) 1837  
# 6 Dashboard 1791  
# 7 Problem Solving 1788  
# 8 Microsoft Excel 1771  
# 9 Sap Applications 1658  
# 10 Operations 1550  
# 11 Project Management 1528  
# 12 Business Process 1484  
  
  
  
# 5 expert, 4 Advanced, 3 Intermediate, 2 Basic Knowledge, 1  
top10\_skills = [  
 "Communication",  
 "Sql (Programming Language)",  
 "Data Analysis",  
 "Management",  
 "Leadership",  
 "Python (Programming Language)",  
 "Dashboard",  
 "Problem Solving",  
 "Microsoft Excel",  
 "Sap Applications"  
]  
  
skills\_data = {  
 "Skill": top10\_skills,  
 "Julio": [3, 4, 4, 4, 3, 3, 4, 4, 5, 4],  
 "Dakota": [4, 4, 4, 3, 5, 3, 5, 4, 5, 3]  
}  
  
df\_team = pd.DataFrame(skills\_data)  
df\_team  
  
  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(8,5))  
sns.heatmap(df\_team.set\_index("Skill"), annot=True, cmap="coolwarm", linewidths=0.5)  
plt.title("Team Skill Levels Heatmap – Julio & Dakota")  
plt.show()  
  
# SKILL GAP (Ideal vs our skills)  
df\_team["Gap\_Julio"] = 5 - df\_team["Julio"]  
df\_team["Gap\_Dakota"] = 5 - df\_team["Dakota"]  
  
# SHOT SKILLS  
print(df\_team[["Skill", "Julio", "Dakota", "Gap\_Julio", "Gap\_Dakota"]])  
  
  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
#Melt AND PLOT GAP  
df\_gaps\_melted = df\_team.melt(  
 id\_vars="Skill",  
 value\_vars=["Gap\_Julio", "Gap\_Dakota"],  
 var\_name="Member",  
 value\_name="Gap"  
)  
  
plt.figure(figsize=(9,5))  
sns.barplot(data=df\_gaps\_melted, x="Skill", y="Gap", hue="Member", palette="coolwarm")  
plt.title("Skill Gaps Comparison – Julio vs Dakota (Ideal = 5)")  
plt.xticks(rotation=45, ha="right")  
plt.ylabel("Gap (Difference from Ideal Level 5)")  
plt.xlabel("Skill")  
plt.legend(title="Team Member")  
plt.tight\_layout()  
plt.show()

0 Merchandising, Mathematics, Presentations, Pre...  
1 Procurement, Ficial Statements, Oracle Busines...  
2 Management, Exception Reporting, Report Writin...  
3 Exit Strategies, Reliability, User Story, Mana...  
4   
Name: ALL\_SKILLS\_RAW, dtype: object  
 Merchandising, Mathematics, Presentations, Predictive Modeling, Data Modeling, Advanced Analytics, Data Extraction, Statistical Analysis, Data Mining, Business Analysis, Fice, Algorithms, Statistics, SQL (Programming Language), Report Writing, Ad Hoc Reporting, Power BI, Relationship Building, Economics, Business Administration Merchandising, Predictive Modeling, Data Modeling, Advanced Analytics, Data Extraction, Statistical Analysis, Data Mining, Business Analysis, Fice, Algorithms, Statistics, SQL (Programming Language), Ad Hoc Reporting, Power BI, Economics SQL (Programming Language), Power BI Mathematics, Presentations, Report Writing, Relationship Building, Business Administration  
 Procurement, Ficial Statements, Oracle Business Intelligence (BI), OBIA, Oracle E-Business Suite, PL, SQL, Supply Chain, Business Intelligence, Oracle Fusion Middleware, Project Accounting Procurement, Ficial Statements, Oracle Business Intelligence (BI), OBIA, Oracle E-Business Suite, PL, SQL, Supply Chain, Business Intelligence, Oracle Fusion Middleware, Project Accounting Oracle Business Intelligence (BI), OBIA, Oracle E-Business Suite, PL, SQL, Oracle Fusion Middleware  
 Management, Exception Reporting, Report Writing, Security Clearance, Interpersonal Communications, Ability To Meet Deadlines, Presentations, Writing, Data Analysis, Organizational Skills, Negotiation, Data Integrity, Microsoft Office Exception Reporting, Data Analysis, Data Integrity Microsoft Office Management, Report Writing, Interpersonal Communications, Ability To Meet Deadlines, Presentations, Writing, Organizational Skills, Negotiation, Microsoft Office Security Clearance  
Exit Strategies, Reliability, User Story, Management, Strategic Planning, Hardware Configuration Management, On Prem, Agile Methodology, Solution Design, Advanced Analytics, Reengineering, Safety Assurance, Cross-Functional Collaboration, Requirements Elicitation, Business Analysis, Data Management, Data Architecture, Influencing Skills, Market Trend, Business Valuation, Creativity, Innovation, Goverce, Systems Development Life Cycle, Leadership, Test Planning, Multi-Tet Cloud Environments, Scrum (Software Development), Project Management, Operations, Data Migration, Regulatory Compliance, Product Roadmaps, SAS (Software), Troubleshooting (Problem Solving), Quality Assurance, Software As A Service (SaaS), Data Domain, Product Requirements, Data Goverce, Competitive Intelligence, Operations Architecture, Risk Appetite, Google Cloud Platform (GCP), User Feedback Exit Strategies, User Story, Hardware Configuration Management, On Prem, Agile Methodology, Solution Design, Advanced Analytics, Reengineering, Cross-Functional Collaboration, Requirements Elicitation, Business Analysis, Data Management, Data Architecture, Market Trend, Business Valuation, Systems Development Life Cycle, Test Planning, Multi-Tet Cloud Environments, Scrum (Software Development), Project Management, Data Migration, Regulatory Compliance, Product Roadmaps, SAS (Software), Software As A Service (SaaS), Data Domain, Product Requirements, Data Goverce, Competitive Intelligence, Operations Architecture, Risk Appetite, Google Cloud Platform (GCP), User Feedback SAS (Software), Google Cloud Platform (GCP) Reliability, Management, Strategic Planning, Safety Assurance, Influencing Skills, Creativity, Innovation, Goverce, Leadership, Operations, Troubleshooting (Problem Solving), Quality Assurance  
   
 Frequency count  
0 Communication 3394  
1 Sql (Programming Language) 3134  
2 Data Analysis 2960  
3 Management 2116  
4 Leadership 2023  
5 Python (Programming Language) 1837  
6 Dashboard 1791  
7 Problem Solving 1788  
8 Microsoft Excel 1771  
9 Sap Applications 1658  
10 Operations 1550  
11 Project Management 1528  
12 Business Process 1484  
13 Fice 1437  
14 Business Requirements 1415  
15 Planning 1211  
16 Presentations 1141  
17 Writing 1120  
18 Detail Oriented 1118  
19 Tableau (Business Intelligence Software) 1116



Skill Julio Dakota Gap\_Julio Gap\_Dakota  
0 Communication 3 4 2 1  
1 Sql (Programming Language) 4 4 1 1  
2 Data Analysis 4 4 1 1  
3 Management 4 3 1 2  
4 Leadership 3 5 2 0  
5 Python (Programming Language) 3 3 2 2  
6 Dashboard 4 5 1 0  
7 Problem Solving 4 4 1 1  
8 Microsoft Excel 5 5 0 0  
9 Sap Applications 4 3 1 2



# Key Takeaways

Julio: Focus on Communication, Leadership, Python, and Management to reduce gaps

Dakota: Focus on Python and Management. Leadership and Dashboard are strong.

## Recommended Actions

Technical Skills: SQL, Python, Dashboarding; Codecademy, DataCamp, Khan Academy, Coursera

SAP Applications: SAP Learning Hub, openSAP Courses

Soft Skills: Communication: LinkedIn Learning, Coursework Leadership and Management: Coursera Management courses

## Team Collaboartion to Bridge Skill Gaps

Cross Training among team members on strong skills, team projects to exercise the weaker skills, and track progress.

# Data Loading and Cleaning

from pyspark.sql import SparkSession  
  
# Start a Spark session  
spark = SparkSession.builder.appName("JobPostingsAnalysis").getOrCreate()  
  
# Load the CSV file into a Spark DataFrame  
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","true").option("escape", "\"").csv("../data/lightcast\_job\_postings.csv")

[Stage 8:> (0 + 1) / 1]

import pandas as pd  
from pyspark.sql.functions import when, col  
  
#Clean Data to convert to Pandas  
columns\_to\_drop = ["ID", "BODY", "URL", "ACTIVE\_URLS", "DUPLICATES", "LAST\_UPDATED\_TIMESTAMP",  
 "NAICS2", "NAICS3", "NAICS4", "NAICS5", "NAICS6",  
 "SOC\_2", "SOC\_3", "SOC\_4", "SOC\_5", "LAST\_UPDATED\_DATE", "LAST\_UPDATED\_TIMESTAMP", "EXPIRED", "SOURCE\_TYPES", "SOURCES", "ACTIVE\_SOURCES\_INFO", "MODELED\_EXPIRED", "MODELED\_DURATION", "NAICS2\_NAME", "NAICS3\_NAME", "NAICS4\_NAME", "NAICS5\_NAME", "NAICS6\_NAME",  
 "SOC\_2\_NAME", "SOC\_3\_NAME", "SOC\_4\_NAME", "SOC\_5\_NAME", "EDUCATION\_LEVELS", "MIN\_EDULEVELS"  
   
]  
cleaned\_data = df.drop(\*columns\_to\_drop)  
  
cleaned\_data = cleaned\_data.withColumn(  
 "REMOTE\_TYPE\_NAME",  
 when(col("REMOTE\_TYPE\_NAME") == "Remote", "Remote")  
 .when(col("REMOTE\_TYPE\_NAME") == "Hybrid Remote", "Hybrid")  
 .when(col("REMOTE\_TYPE\_NAME") == "[None]", "On-site")  
 .when(col("REMOTE\_TYPE\_NAME").isNull(), "On-site")  
 .when(col("REMOTE\_TYPE\_NAME") == "Not Remote", "On-site")  
 .otherwise(col("REMOTE\_TYPE\_NAME"))  
)  
  
cleaned\_data = cleaned\_data.withColumn(  
 "EMPLOYMENT\_TYPE\_NAME",  
 when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time / full-time", "Flexible")  
 .when(col("EMPLOYMENT\_TYPE\_NAME").isNull(), "Full-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time (â‰¤ 32 hours)", "Part-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Full-time (> 32 hours)", "Full-Time")  
 .otherwise(col("EMPLOYMENT\_TYPE\_NAME"))   
)  
  
cleaned\_data = cleaned\_data.filter(col("NAICS\_2022\_2\_NAME") != "Unclassified Industry")  
  
median\_salary = cleaned\_data.approxQuantile("SALARY", [0.5], 0.01)[0]  
cleaned\_data = cleaned\_data.withColumn(  
 "SALARY",  
 when(col("SALARY").isNull(), median\_salary).otherwise(col("SALARY"))  
)  
  
#Convert to Pandas  
clean\_pdf = cleaned\_data.toPandas()

[Stage 9:> (0 + 1) / 1] [Stage 10:> (0 + 1) / 1]

# Cleaning empty rows and dropping columns that are mostly empty  
  
  
fill\_cols = ["CITY\_NAME", "CITY", "LOCATION", "STATE", "STATE\_NAME", "COMPANY", "COMPANY\_NAME"]  
clean\_pdf[fill\_cols] = clean\_pdf[fill\_cols].fillna("Unknown")  
  
clean\_pdf = clean\_pdf.drop\_duplicates(subset=["TITLE", "COMPANY", "LOCATION", "POSTED"], keep="first")  
  
clean\_pdf.dropna(thresh=len(clean\_pdf)\*0.5, axis=1, inplace=True)  
  
#New Column to Classify AI Jobs and Add Month of Posting Date  
  
ai\_keywords = [  
 "AI", "Machine Learning", "Data Scientist", "Data Analyst", "ML",   
 "Artificial Intelligence", "Deep Learning", "NLP", "Predictive Analytics"  
]  
  
#Function to classify AI vs Non-AI Jobs  
def classify\_ai(title):  
 title\_lower = str(title).lower()  
 for keyword in ai\_keywords:  
 if keyword.lower() in title\_lower:  
 return "AI"  
 return "Non-AI"  
  
clean\_pdf["AI\_JOB"] = clean\_pdf["TITLE\_RAW"].apply(classify\_ai)  
  
clean\_pdf["POSTED"] = pd.to\_datetime(clean\_pdf["POSTED"], errors="coerce")  
clean\_pdf["POSTED\_MONTH"] = clean\_pdf["POSTED"].dt.month  
  
#Add column for URBAN vs RURAL  
  
clean\_pdf["URBAN\_RURAL"] = clean\_pdf["MSA\_NAME"].apply(lambda x: "Urban" if pd.notnull(x) else "Rural")

# Model 1: KMeans Clustering

We applied KMeans clustering to identify patterns in the data. We wanted to each cluster to represent a group of jobs based on characteristics of Salary, AI jobs, remote work, and Urban or Rural locations. We used an elbow method to find the ideal number of clusters, which was 4, though the model didn’t reflect as concise groupings as wanted.

Features:

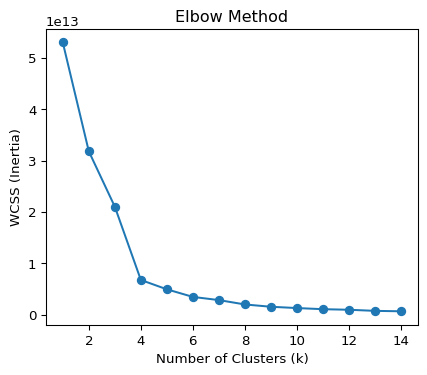
SALARY: Continuous numerical feature to reflect compensation. AI\_JOB: Binary feature (AI vs Non-AI), one-hot encoded. REMOTE\_TYPE\_NAME: One-hot encoded categories (On-site, Remote, Hybrid). URBAN\_RURAL: One-hot encoded (Urban or Rural).

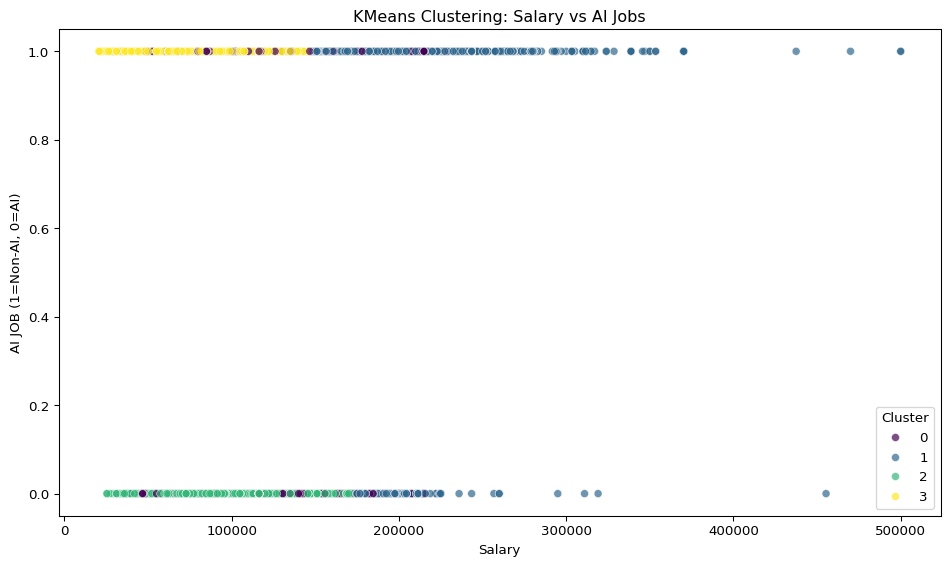
Implications for Job Seekers

A job seeker can use this model to see what type of characteristics of a job might be tied to others. For example, AI jobs might yield high salaries, and if we used a reference of industry might be able to find an industry of interest that falls in a cluster that is being regarded in the job hunt. This can also be tied into the Skills Gap Analysis to see what a job seeker should work on in order to be considered for a certain cluster.

#KMeans clustering using NAICS as a reference but not a target  
  
from sklearn.preprocessing import StandardScaler  
  
# Select features  
features = clean\_pdf[["SALARY", "AI\_JOB", "REMOTE\_TYPE\_NAME", "URBAN\_RURAL"]]  
  
# One-hot encode categorical columns  
features\_encoded = pd.get\_dummies(features, columns=["AI\_JOB", "REMOTE\_TYPE\_NAME", "URBAN\_RURAL"], drop\_first=True)  
  
# Standardize numerical features (important for KMeans)  
scaler = StandardScaler()  
features\_scaled = scaler.fit\_transform(features\_encoded)  
  
from sklearn.cluster import KMeans  
  
k = 4  
kmeans = KMeans(n\_clusters=k, random\_state=42)  
clean\_pdf["CLUSTER"] = kmeans.fit\_predict(features\_scaled)  
  
#Use Industry Name (NAICS2022) as a reference label  
  
cluster\_summary = (  
 clean\_pdf.groupby(["CLUSTER", "NAICS\_2022\_2\_NAME"])  
 .size()  
 .reset\_index(name="count")  
 .sort\_values(["CLUSTER", "count"], ascending=[True, False])  
)  
  
print(cluster\_summary)  
  
#Used an Elbow Method to choose the correct number of clusters  
  
from sklearn.cluster import KMeans  
import matplotlib.pyplot as plt  
  
# Example features  
X = features\_encoded.values # your numerical features  
  
wcss = []  
for k in range(1, 15):  
 km = KMeans(n\_clusters=k, random\_state=42)  
 km.fit(X)  
 wcss.append(km.inertia\_)  
  
plt.plot(range(1, 15), wcss, marker='o')  
plt.xlabel('Number of Clusters (k)')  
plt.ylabel('WCSS (Inertia)')  
plt.title('Elbow Method')  
plt.show()  
  
cluster\_summary.head(20) # Show top 20 to see patterns  
  
one\_hot\_cols = ['AI\_JOB\_Non-AI', 'REMOTE\_TYPE\_NAME\_On-site', 'REMOTE\_TYPE\_NAME\_Remote', 'URBAN\_RURAL\_Urban']  
  
clean\_pdf = pd.concat([clean\_pdf, features\_encoded[one\_hot\_cols]], axis=1)  
  
  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(10,6))  
sns.scatterplot(  
 data=clean\_pdf,  
 x='SALARY',  
 y='AI\_JOB\_Non-AI',   
 hue='CLUSTER',  
 palette='viridis',  
 alpha=0.7  
)  
plt.title("KMeans Clustering: Salary vs AI Jobs")  
plt.xlabel("Salary")  
plt.ylabel("AI JOB (1=Non-AI, 0=AI)")  
plt.legend(title="Cluster")  
plt.tight\_layout()  
plt.show()

CLUSTER \  
13 0   
6 0   
1 0   
10 0   
8 0   
.. ...   
60 3   
69 3   
71 3   
63 3   
62 3   
  
 NAICS\_2022\_2\_NAME \  
13 Professional, Scientific, and Technical Services   
6 Finance and Insurance   
1 Administrative and Support and Waste Management and Remediation Services   
10 Manufacturing   
8 Information   
.. ...   
60 Accommodation and Food Services   
69 Management of Companies and Enterprises   
71 Mining, Quarrying, and Oil and Gas Extraction   
63 Arts, Entertainment, and Recreation   
62 Agriculture, Forestry, Fishing and Hunting   
  
 count   
13 2968   
6 1720   
1 1598   
10 734   
8 696   
.. ...   
60 353   
69 146   
71 95   
63 71   
62 45   
  
[80 rows x 3 columns]

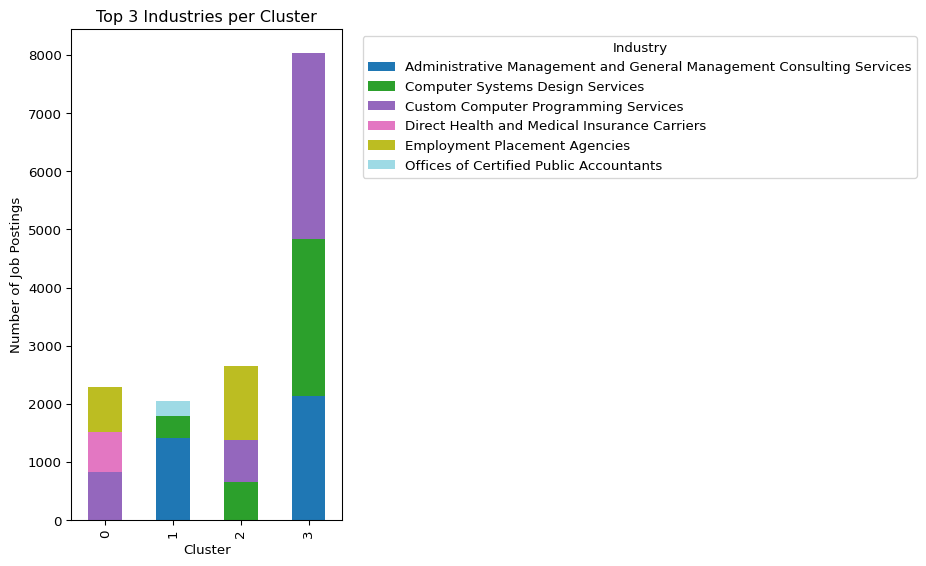




#Visualizing the NAICS Industries in reference to each cluster  
  
  
  
import matplotlib.pyplot as plt  
  
  
top\_industries = (  
 clean\_pdf.groupby(["CLUSTER", "NAICS\_2022\_6\_NAME"])  
 .size()  
 .reset\_index(name="count")  
)  
  
top3 = top\_industries.groupby("CLUSTER").apply(lambda x: x.nlargest(3, "count")).reset\_index(drop=True)  
  
  
pivot\_df = top3.pivot(index="CLUSTER", columns="NAICS\_2022\_6\_NAME", values="count").fillna(0)  
  
print(pivot\_df)  
  
pivot\_df.plot(kind='bar', stacked=True, figsize=(10,6), colormap='tab20')   
  
plt.title("Top 3 Industries per Cluster")  
plt.xlabel("Cluster")  
plt.ylabel("Number of Job Postings")  
plt.legend(title="Industry", bbox\_to\_anchor=(1.05, 1), loc='upper left')  
plt.tight\_layout()  
plt.show()

/tmp/ipykernel\_7310/24634743.py:14: FutureWarning:  
  
DataFrameGroupBy.apply operated on the grouping columns. This behavior is deprecated, and in a future version of pandas the grouping columns will be excluded from the operation. Either pass `include\_groups=False` to exclude the groupings or explicitly select the grouping columns after groupby to silence this warning.

NAICS\_2022\_6\_NAME Administrative Management and General Management Consulting Services \  
CLUSTER   
0 0.0   
1 1414.0   
2 0.0   
3 2139.0   
  
NAICS\_2022\_6\_NAME Computer Systems Design Services \  
CLUSTER   
0 0.0   
1 376.0   
2 663.0   
3 2694.0   
  
NAICS\_2022\_6\_NAME Custom Computer Programming Services \  
CLUSTER   
0 836.0   
1 0.0   
2 720.0   
3 3209.0   
  
NAICS\_2022\_6\_NAME Direct Health and Medical Insurance Carriers \  
CLUSTER   
0 685.0   
1 0.0   
2 0.0   
3 0.0   
  
NAICS\_2022\_6\_NAME Employment Placement Agencies \  
CLUSTER   
0 774.0   
1 0.0   
2 1268.0   
3 0.0   
  
NAICS\_2022\_6\_NAME Offices of Certified Public Accountants   
CLUSTER   
0 0.0   
1 257.0   
2 0.0   
3 0.0



# Key Insights

Out of the 4 clusters, it seems like there are 3 fairly clear clusters and 1 that is more ambiguous.

Cluster 0: The ambiguous one, doesn’t seem to have any groupings along AI, Non-AI, and Salary. Cluster 1: This is a group of higher salary paid positions, but are grouped regardless of AI vs Non-AI. Cluster 2: These seem to be low paying, AI jobs. Cluster 3: These seem to be low paying Non-AI jobs.

Industry Relationship:

It seems that the industries don’t have as much of a bearing on each cluster. For example, we thought that the high paying clusters would be more tech industry focused, but the low paying cluster also has a large amount of job openings in the same industries. This may mean that salaries are most likely influenced less by Industry and AI impact and influenced more by Skills, seniority, and education.

# Model 2: Linear Regression for Salary Prediction

We ran a linear regression model to predict job salaries based on location, remote status, and urban/rural classification. It estimates/predicts how location and job type impact salary. It’s useful for identifying which factors contribute to higher salaries and preferred work type.

Features used:

STATE\_NAME: One-hot encoded categorical variable representing each state. REMOTE\_TYPE\_NAME: One-hot encoded (Remote, Hybrid, On-site). URBAN\_RURAL: One-hot encoded (Urban vs. Rural). Target variable: SALARY (numerical).

Implications for job seekers:

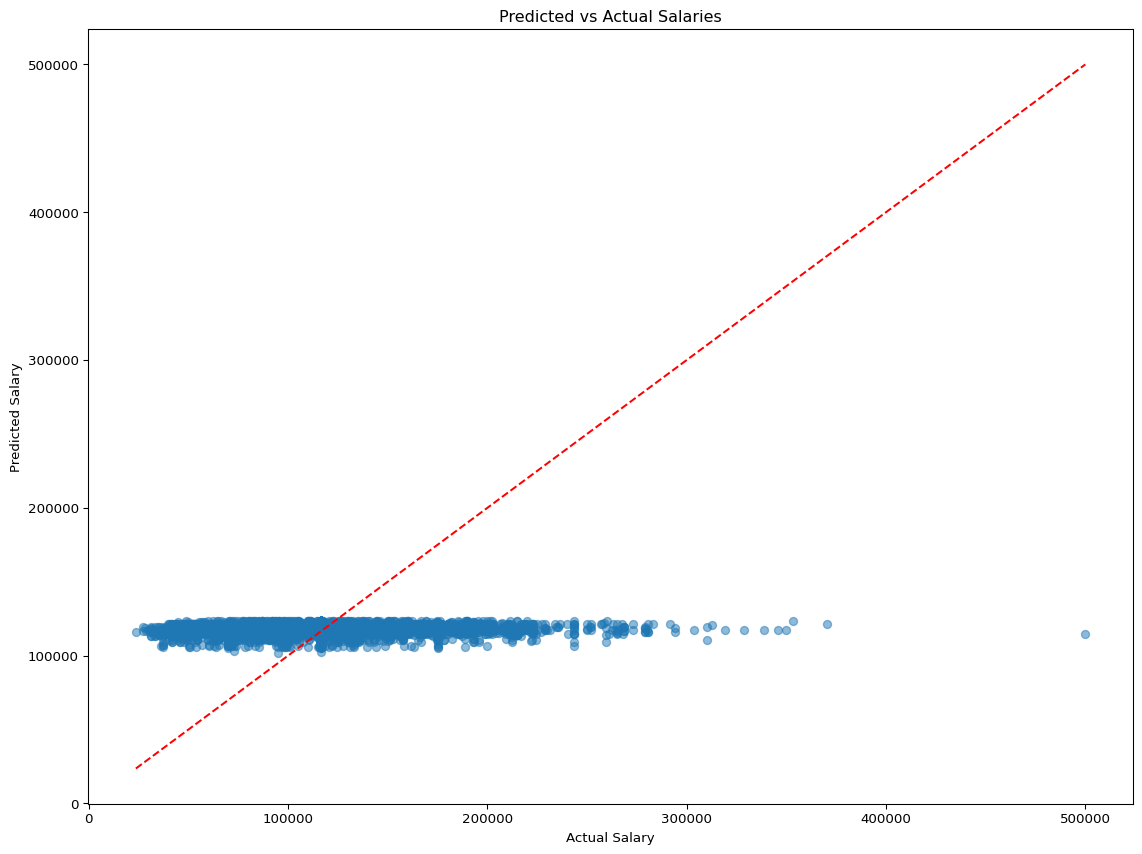
The model reveals which locations and job types may pay more, which is probably the most important consideration for job seekers.

For example, remote AI jobs in urban hubs may offer higher salaries than non-AI roles in rural areas.

Limitations: This model focuses only on geographic and job-type features, so salary effects of skills, experience, or certifications are not captured. As we show, geographical data is not a greate predictor or estimator in Salary. We assume that education levels, experience, and skills may be a better predictor.

#Predicting Salaries based on Location Data through Linear Regression  
#\*Decided to run it in Pandas with Scikit as it's already been converted and cleaned  
  
from sklearn.linear\_model import LinearRegression  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import mean\_squared\_error, r2\_score  
import numpy as np  
import matplotlib.pyplot as plt  
  
reg\_data = clean\_pdf[["SALARY", "STATE\_NAME", "REMOTE\_TYPE\_NAME", "URBAN\_RURAL"]]  
  
X = pd.get\_dummies(reg\_data[["STATE\_NAME", "REMOTE\_TYPE\_NAME", "URBAN\_RURAL"]], drop\_first=True)  
y = reg\_data["SALARY"]  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=.2, random\_state=42)  
  
model = LinearRegression()  
model.fit(X\_train, y\_train)  
  
y\_pred = model.predict(X\_test)  
rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))  
r2 = r2\_score(y\_test, y\_pred)  
  
print(f"Evaulation Metrics:")  
print(f"RMSE: {rmse:,.2f}")  
print(f"R2: {r2:.3f}")  
  
df\_coef = pd.DataFrame({  
 "Feature": X.columns,  
 "Coefficient": model.coef\_  
}).sort\_values(by="Coefficient", ascending=False)  
  
print("\nTop PositiveInfluences on Salary:")  
print(df\_coef.head(10))  
  
print("\nTop Negative Influences on Salary:")  
print(df\_coef.tail(10))  
  
  
plt.figure(figsize=(12,9))  
plt.scatter(y\_test, y\_pred, alpha=0.5)  
plt.xlabel("Actual Salary")  
plt.ylabel("Predicted Salary")  
plt.title("Predicted vs Actual Salaries")  
plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--')  
plt.tight\_layout()  
plt.show()

Evaulation Metrics:  
RMSE: 29,689.15  
R2: 0.006  
  
Top PositiveInfluences on Salary:  
 Feature Coefficient  
45 STATE\_NAME\_Washington 6361.179175  
5 STATE\_NAME\_Connecticut 5837.080401  
43 STATE\_NAME\_Vermont 5302.264696  
3 STATE\_NAME\_California 4726.943831  
51 REMOTE\_TYPE\_NAME\_Remote 4274.612550  
50 REMOTE\_TYPE\_NAME\_On-site 4176.813834  
2 STATE\_NAME\_Arkansas 4109.440464  
46 STATE\_NAME\_Washington, D.C. (District of Columbia) 2818.294899  
28 STATE\_NAME\_New Jersey 2786.906817  
11 STATE\_NAME\_Illinois 2575.153394  
  
Top Negative Influences on Salary:  
 Feature Coefficient  
49 STATE\_NAME\_Wyoming -3610.797563  
47 STATE\_NAME\_West Virginia -3985.698913  
22 STATE\_NAME\_Mississippi -4039.741194  
15 STATE\_NAME\_Kentucky -5710.215799  
0 STATE\_NAME\_Alaska -5941.167716  
42 STATE\_NAME\_Utah -6390.023354  
26 STATE\_NAME\_Nevada -7679.185737  
39 STATE\_NAME\_South Dakota -9360.455547  
32 STATE\_NAME\_North Dakota -10354.011369  
29 STATE\_NAME\_New Mexico -10744.064282



# Key Insights

Evaulation Metrics: RMSE: 29,689.15 R2: 0.006

These metrics are saying that our features are not very influential on the salary, meaning that there are most likely better predictors of Salary than geographical predictors. Like previously stated, things like seniority, education, and skills may be better predictors to higher or lower salaries.

Another insight that might be able to be used by a job seeker might be the highest and negative influencers on salary. Even though our model isn’t great, it seems that we can deduce that seeking a job in Washington might yield a higher salary vs. the baseline salary, but seeking a job in New Mexico might yield a lower salary than the baseline.