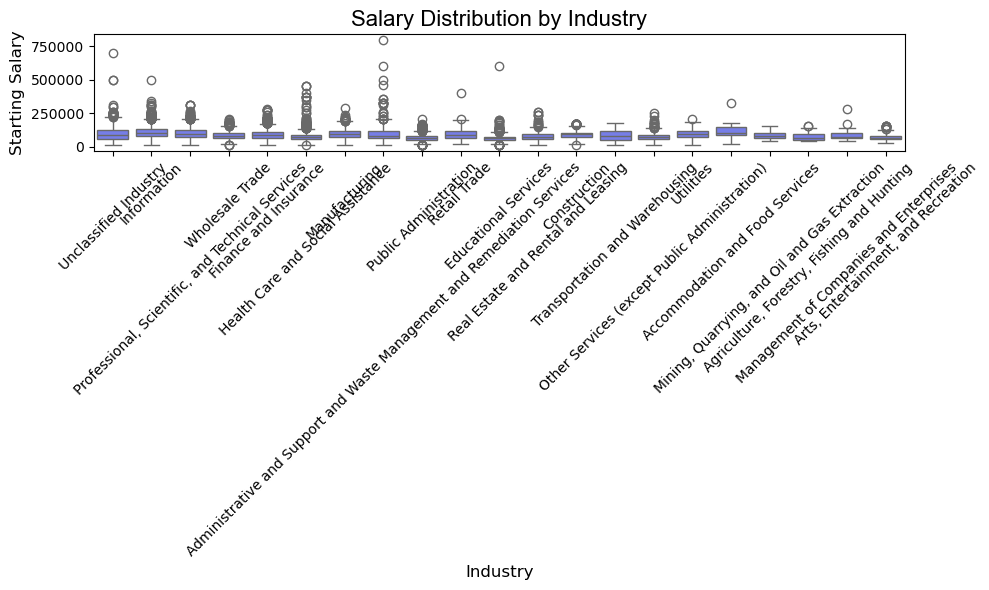
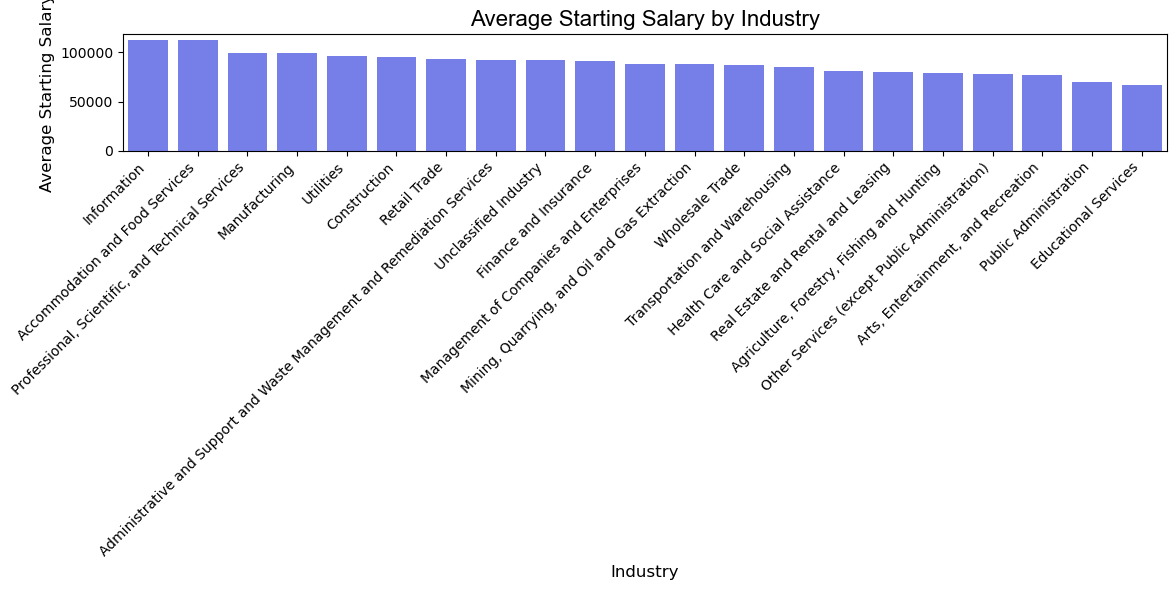
Remote Vs On-site Job Posting

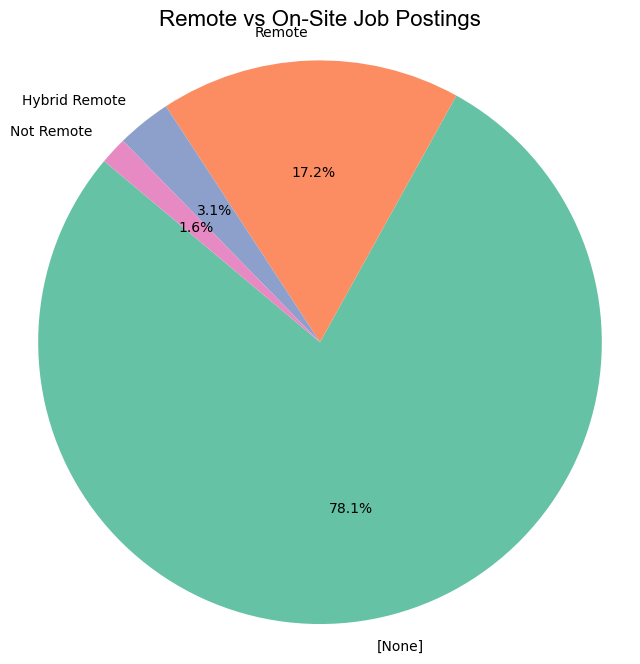
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
# Load dataset (correct path syntax)  
data = pd.read\_csv(r"C:\Users\pooja\Desktop\Repositories\ad688-employability-sp25A1-pooja\lightcast\_job\_postings.csv")  
  
# Drop any missing values just in case  
pdf = data[["NAICS2\_NAME", "SALARY\_FROM"]].dropna()  
  
# Plot  
plt.figure(figsize=(10, 6))  
sns.boxplot(data=pdf, x="NAICS2\_NAME", y="SALARY\_FROM", color="#636EFA")  
  
# Add labels and title  
plt.title("Salary Distribution by Industry", fontsize=16, fontname="Arial")  
plt.xlabel("Industry", fontsize=12)  
plt.ylabel("Starting Salary", fontsize=12)  
plt.xticks(rotation=45)  
plt.tight\_layout()  
  
# Show plot  
plt.show()



import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
# Load dataset (correct path syntax)  
data = pd.read\_csv(r"C:\Users\pooja\Desktop\Repositories\ad688-employability-sp25A1-pooja\lightcast\_job\_postings.csv")  
  
# Drop missing values  
pdf = data[["NAICS2\_NAME", "SALARY\_FROM"]].dropna()  
  
# Group by industry and calculate average salary  
avg\_salary = pdf.groupby("NAICS2\_NAME", as\_index=False)["SALARY\_FROM"].mean()  
  
# Sort industries by average salary (optional)  
avg\_salary = avg\_salary.sort\_values(by="SALARY\_FROM", ascending=False)  
  
# Plot  
plt.figure(figsize=(12, 6))  
sns.barplot(data=avg\_salary, x="NAICS2\_NAME", y="SALARY\_FROM", color="#636EFA")  
  
# Customize  
plt.title("Average Starting Salary by Industry", fontsize=16, fontname="Arial")  
plt.xlabel("Industry", fontsize=12)  
plt.ylabel("Average Starting Salary", fontsize=12)  
plt.xticks(rotation=45, ha='right')  
plt.tight\_layout()  
  
# Show plot  
plt.show()



import pandas as pd  
import matplotlib.pyplot as plt  
  
# Step 1: Load the dataset (update path as needed)  
df = pd.read\_csv(r"C:\Users\pooja\Desktop\Repositories\ad688-employability-sp25A1-pooja\lightcast\_job\_postings.csv")  
  
# Step 2: Select and clean the relevant column  
pdf = df[["REMOTE\_TYPE\_NAME"]].dropna()  
  
# Step 3: Count occurrences of each remote type  
remote\_type\_counts = pdf["REMOTE\_TYPE\_NAME"].value\_counts().reset\_index()  
remote\_type\_counts.columns = ["REMOTE\_TYPE\_NAME", "Job Count"]  
  
# Step 4: Create pie chart using matplotlib  
plt.figure(figsize=(8, 8))  
plt.pie(  
 remote\_type\_counts["Job Count"],  
 labels=remote\_type\_counts["REMOTE\_TYPE\_NAME"],  
 autopct='%1.1f%%',  
 startangle=140,  
 colors=plt.cm.Set2.colors # Optional color palette  
)  
  
# Step 5: Add title and format  
plt.title("Remote vs On-Site Job Postings", fontsize=16, fontname="Arial")  
plt.axis("equal") # Ensures the pie chart is circular  
  
# Step 6: Show plot  
plt.show()



df.columns

Index(['ID', 'LAST\_UPDATED\_DATE', 'LAST\_UPDATED\_TIMESTAMP', 'DUPLICATES',  
 'POSTED', 'EXPIRED', 'DURATION', 'SOURCE\_TYPES', 'SOURCES', 'URL',  
 ...  
 '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', 'NAICS\_2022\_6',  
 'NAICS\_2022\_6\_NAME'],  
 dtype='object', length=131)

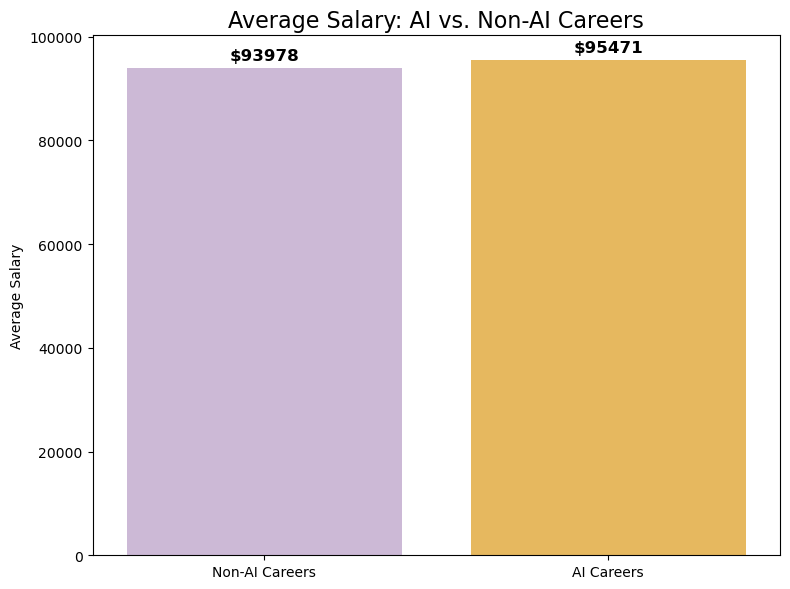
column\_list = df.columns.tolist()  
print(column\_list)

['ID', 'LAST\_UPDATED\_DATE', 'LAST\_UPDATED\_TIMESTAMP', 'DUPLICATES', 'POSTED', 'EXPIRED', 'DURATION', 'SOURCE\_TYPES', 'SOURCES', 'URL', 'ACTIVE\_URLS', 'ACTIVE\_SOURCES\_INFO', 'TITLE\_RAW', 'BODY', 'MODELED\_EXPIRED', 'MODELED\_DURATION', 'COMPANY', 'COMPANY\_NAME', 'COMPANY\_RAW', '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', 'CITY', 'CITY\_NAME', 'COUNTY', 'COUNTY\_NAME', 'MSA', 'MSA\_NAME', 'STATE', 'STATE\_NAME', 'COUNTY\_OUTGOING', 'COUNTY\_NAME\_OUTGOING', 'COUNTY\_INCOMING', 'COUNTY\_NAME\_INCOMING', 'MSA\_OUTGOING', 'MSA\_NAME\_OUTGOING', 'MSA\_INCOMING', 'MSA\_NAME\_INCOMING', 'NAICS2', 'NAICS2\_NAME', 'NAICS3', 'NAICS3\_NAME', 'NAICS4', 'NAICS4\_NAME', 'NAICS5', 'NAICS5\_NAME', 'NAICS6', 'NAICS6\_NAME', 'TITLE', 'TITLE\_NAME', 'TITLE\_CLEAN', 'SKILLS', 'SKILLS\_NAME', 'SPECIALIZED\_SKILLS', 'SPECIALIZED\_SKILLS\_NAME', 'CERTIFICATIONS', 'CERTIFICATIONS\_NAME', 'COMMON\_SKILLS', 'COMMON\_SKILLS\_NAME', 'SOFTWARE\_SKILLS', 'SOFTWARE\_SKILLS\_NAME', 'ONET', 'ONET\_NAME', 'ONET\_2019', 'ONET\_2019\_NAME', 'CIP6', 'CIP6\_NAME', 'CIP4', 'CIP4\_NAME', 'CIP2', 'CIP2\_NAME', 'SOC\_2021\_2', 'SOC\_2021\_2\_NAME', 'SOC\_2021\_3', 'SOC\_2021\_3\_NAME', 'SOC\_2021\_4', 'SOC\_2021\_4\_NAME', 'SOC\_2021\_5', 'SOC\_2021\_5\_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', 'SOC\_2', 'SOC\_2\_NAME', 'SOC\_3', 'SOC\_3\_NAME', 'SOC\_4', 'SOC\_4\_NAME', 'SOC\_5', 'SOC\_5\_NAME', 'LIGHTCAST\_SECTORS', 'LIGHTCAST\_SECTORS\_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', 'NAICS\_2022\_6', 'NAICS\_2022\_6\_NAME']

# Bar Plot – Average Salary: AI vs. Non-AI Career

import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
df = pd.read\_csv(r"C:\Users\pooja\Desktop\Repositories\ad688-employability-sp25A1-pooja\lightcast\_job\_postings.csv")  
  
# Filter AI-related jobs  
df['IS\_AI\_JOB'] = df['TITLE\_NAME'].str.contains("AI|Artificial Intelligence|Machine Learning|Data Scientist", case=False, na=False)  
  
# Group by AI vs. Non-AI and calculate average salary  
salary\_compare = df[df["SALARY\_FROM"].notna()].groupby("IS\_AI\_JOB")["SALARY\_FROM"].mean().reset\_index()  
salary\_compare["Category"] = salary\_compare["IS\_AI\_JOB"].map({True: "AI Careers", False: "Non-AI Careers"})  
  
# Plot  
plt.figure(figsize=(8, 6))  
barplot = sns.barplot(data=salary\_compare, x="Category", y="SALARY\_FROM", palette=["#cdb4db", "#fcbf49"])  
  
# Add numbers on top of bars  
for i, row in salary\_compare.iterrows():  
 barplot.text(  
 i,   
 row["SALARY\_FROM"] + 1000, # Slightly above the bar  
 f"${row['SALARY\_FROM']:.0f}",   
 ha='center',   
 va='bottom',   
 fontsize=12,   
 fontweight='bold'  
 )  
  
# Customize plot  
plt.title("Average Salary: AI vs. Non-AI Careers", fontsize=16)  
plt.ylabel("Average Salary")  
plt.xlabel("")  
plt.tight\_layout()  
plt.show()

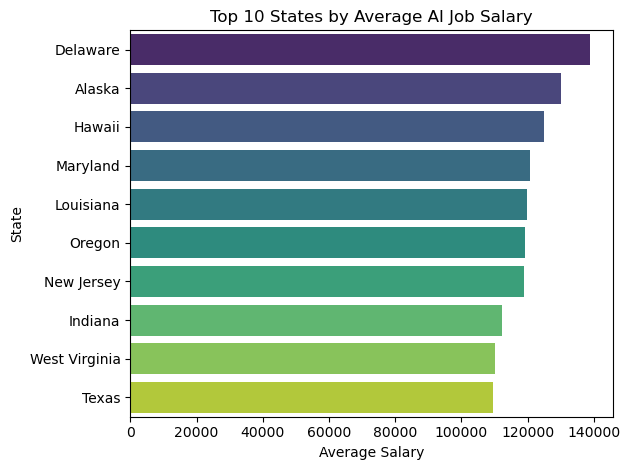
C:\Users\pooja\AppData\Local\Temp\ipykernel\_61152\3999385079.py:16: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.  
  
 barplot = sns.barplot(data=salary\_compare, x="Category", y="SALARY\_FROM", palette=["#cdb4db", "#fcbf49"])



# Bar Plot – Top Regions by AI Salary

# Filter AI jobs and drop missing salary/state  
ai\_jobs = df[df["IS\_AI\_JOB"] & df["SALARY\_FROM"].notna() & df["STATE\_NAME"].notna()]  
  
# Group and get top 10 states by average salary  
top\_states = ai\_jobs.groupby("STATE\_NAME")["SALARY\_FROM"].mean().sort\_values(ascending=False).head(10).reset\_index()  
  
# Plot  
sns.barplot(data=top\_states, x="SALARY\_FROM", y="STATE\_NAME", palette="viridis")  
plt.title("Top 10 States by Average AI Job Salary")  
plt.xlabel("Average Salary")  
plt.ylabel("State")  
plt.tight\_layout()  
plt.show()

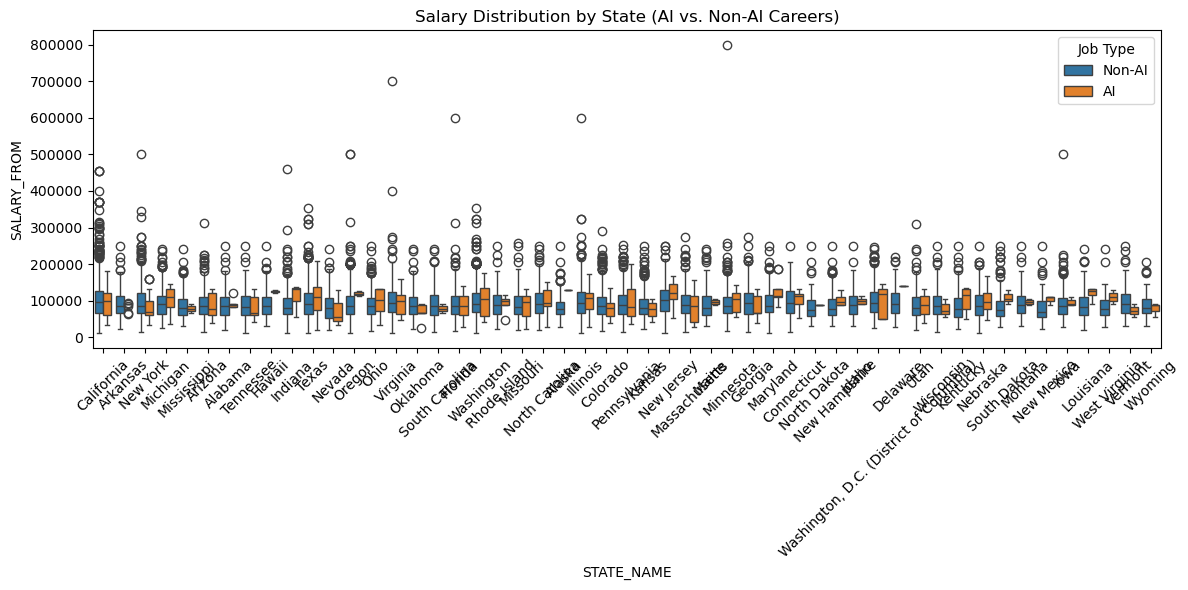
C:\Users\pooja\AppData\Local\Temp\ipykernel\_61152\1523654589.py:8: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.barplot(data=top\_states, x="SALARY\_FROM", y="STATE\_NAME", palette="viridis")



# Box Plot – Salary Distribution by Region (AI vs. Non-AI)

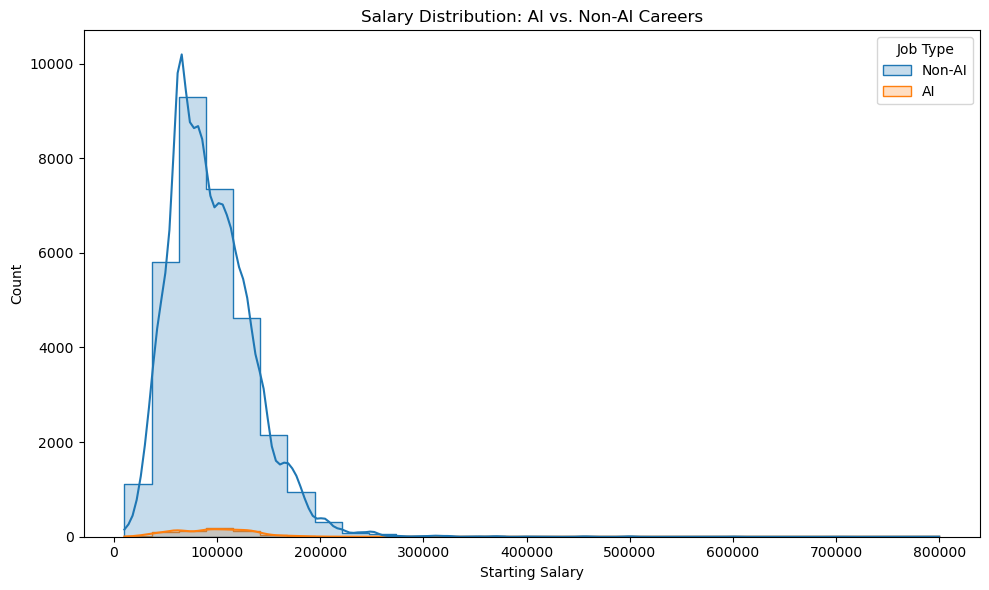
df\_filtered = df[df["SALARY\_FROM"].notna() & df["STATE\_NAME"].notna()]  
df\_filtered["Job Type"] = df\_filtered["IS\_AI\_JOB"].map({True: "AI", False: "Non-AI"})  
  
plt.figure(figsize=(12, 6))  
sns.boxplot(data=df\_filtered, x="STATE\_NAME", y="SALARY\_FROM", hue="Job Type")  
plt.xticks(rotation=45)  
plt.title("Salary Distribution by State (AI vs. Non-AI Careers)")  
plt.tight\_layout()  
plt.show()

C:\Users\pooja\AppData\Local\Temp\ipykernel\_61152\4042802457.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  
 df\_filtered["Job Type"] = df\_filtered["IS\_AI\_JOB"].map({True: "AI", False: "Non-AI"})



# Histogram – Salary Range Distribution (AI vs. Non-AI)

plt.figure(figsize=(10, 6))  
sns.histplot(data=df\_filtered, x="SALARY\_FROM", hue="Job Type", bins=30, kde=True, element="step")  
plt.title("Salary Distribution: AI vs. Non-AI Careers")  
plt.xlabel("Starting Salary")  
plt.tight\_layout()  
plt.show()



# Pie Chart – Proportion of AI vs. Non-AI Jobs in Dataset

job\_counts = df["IS\_AI\_JOB"].value\_counts().reset\_index()  
job\_counts.columns = ["IS\_AI\_JOB", "Count"]  
job\_counts["Category"] = job\_counts["IS\_AI\_JOB"].map({True: "AI Careers", False: "Non-AI Careers"})  
  
plt.pie(job\_counts["Count"], labels=job\_counts["Category"], autopct='%1.1f%%', startangle=140, colors=plt.cm.Set2.colors)  
plt.axis("equal")  
plt.title("AI vs. Non-AI Job Proportion")  
plt.show()

