Exploratory Data Analysis (EDA)

Visual and Statistical Insights from Job Market Data

Anu Sharma, Cindy Guzman, Gavin Boss

2025-10-09

### Overview

This Exploratory Data Analysis (EDA) examines job postings from Lightcast to uncover salary trends, experience requirements, remote work dynamics, and skill demand across Business Analytics (BA), Data Science (DS), and Machine Learning (ML) roles. Lightcast is a leading labor market analytics provider, whose datasets are widely used to study real-time hiring trends and skills demand (Lightcast (2024)). The analysis prepares the dataset for modeling and provides statistical and visual insights that guide the regression and feature engineering stages of the project.

Specifically, this section covers:

* Data preparation and cleaning – handling duplicates, renaming key columns, converting data types, and computing an Average\_Salary variable
* Salary distribution and outliers – visualizing the spread of compensation and identifying high-paying ML and senior roles
* Experience and salary relationships – analyzing how required years of experience influence pay across remote and hybrid work types
* Role-based salary comparison – comparing median salaries for BA, DS, and ML roles
* Remote work trends – exploring how work flexibility impacts compensation
* Top skill frequencies – identifying the most in-demand technical and analytical skills in job descriptions
* Feature correlations – assessing relationships between numeric variables such as salary and experience to inform model selection

Together, these insights establish a clear understanding of labor market patterns and help define which features are most predictive for the upcoming salary regression models.

### Load and Prepare Data

import pandas as pd  
import numpy as np  
import plotly.express as px  
import plotly.graph\_objects as go  
from plotly.subplots import make\_subplots  
import plotly.figure\_factory as ff  
import plotly.io as pio  
import plotly.colors as pc  
import json  
import re  
from collections import Counter  
  
pio.templates.default = "plotly\_white"  
pio.renderers.default = "iframe\_connected"  
  
# === Load data from CSV ===  
df = pd.read\_csv("data/lightcast\_job\_postings.csv", low\_memory=False)  
# print(f"Dataset loaded: {len(df):,} rows, {len(df.columns)} columns")  
  
# --- Detect & drop duplicate columns ---  
# --- Detect & fully clean duplicate-like columns ---  
# Normalize column names: strip whitespace and hidden characters  
df.columns = df.columns.str.strip().str.replace(r"\s+", " ", regex=True)  
  
# Collapse exact duplicates after cleanup  
before\_cols = len(df.columns)  
df = df.loc[:, ~df.columns.duplicated()]  
after\_cols = len(df.columns)  
  
# print(f" Cleaned column names: removed {before\_cols - after\_cols} duplicate(s).")  
# print("Unique columns now:", len(df.columns))  
  
# --- Convert numeric columns safely ---  
for col in ["SALARY\_FROM", "SALARY\_TO", "MIN\_YEARS\_EXPERIENCE", "MAX\_YEARS\_EXPERIENCE"]:  
 if col in df.columns:  
 df[col] = pd.to\_numeric(df[col], errors="coerce")  
  
# --- Compute average salary (avoid string concat) ---  
if {"SALARY\_FROM", "SALARY\_TO"}.issubset(df.columns):  
 df["Average\_Salary"] = df[["SALARY\_FROM", "SALARY\_TO"]].mean(axis=1, skipna=True)  
  
# --- Clean and rename (safely) ---  
rename\_map = {  
 "REMOTE\_TYPE\_NAME": "REMOTE\_GROUP",  
 "STATE\_NAME": "STATE",  
 "LOT\_V6\_OCCUPATION\_GROUP\_NAME": "ROLE\_GROUP"  
}  
  
# Only rename columns that won't create duplicates  
for old, new in rename\_map.items():  
 if old in df.columns and new not in df.columns:  
 df.rename(columns={old: new}, inplace=True)  
 elif old in df.columns and new in df.columns:  
 # print(f" Skipping rename '{old}' → '{new}' to avoid duplicate column name.")  
 pass  
  
  
# --- Drop invalid rows early ---  
if "Average\_Salary" in df.columns:  
 df = df[df["Average\_Salary"].notna() & (df["Average\_Salary"] > 0)]  
  
# --- Downsample if dataset is large ---  
if len(df) > 5000:  
 df = df.sample(5000, random\_state=42)  
  
# print(f" Loaded {len(df)} rows safely with {len(df.columns)} unique columns.")

### Data Preparation and Cleaning

The dataset used for this analysis was sourced from Lightcast job postings, containing thousands of listings across Business Analytics (BA), Data Science (DS), and Machine Learning (ML) roles. To ensure data quality and consistency:

Column Normalization: Extra spaces and hidden characters were stripped from column names to avoid mismatches and duplicates.

Duplicate Removal: Identical columns were collapsed to retain only unique fields.

Type Conversion: Numeric columns such as SALARY\_FROM, SALARY\_TO, MIN\_YEARS\_EXPERIENCE, and MAX\_YEARS\_EXPERIENCE were coerced into numeric format, with non-numeric values safely converted to NaN.

Average Salary Calculation: A new feature, Average\_Salary, was computed as the mean of the salary range for each posting to simplify analysis.

Column Renaming: Key columns were standardized (REMOTE\_TYPE\_NAME → REMOTE\_GROUP, STATE\_NAME → STATE, LOT\_V6\_OCCUPATION\_GROUP\_NAME → ROLE\_GROUP) for clarity.

Invalid Data Removal: Rows with missing or non-positive salaries were dropped.

Downsampling: For performance, the dataset was randomly reduced to 5,000 representative rows, preserving the statistical diversity of the original population.

This preprocessing established a clean, consistent dataset suitable for visualization and modeling.

# Filter valid salaries  
df\_salary = df[df["Average\_Salary"] > 0]  
  
# --- Base histogram (same as your original) ---  
fig = px.histogram(  
 df\_salary,  
 x="Average\_Salary",  
 nbins=40,  
 color\_discrete\_sequence=["#B03A73"],   
 title="Distribution of Average Salaries"  
)  
  
# --- Compute log overlay using numpy ---  
log\_salaries = np.log(df\_salary["Average\_Salary"])  
counts, bins = np.histogram(log\_salaries, bins=40)  
bin\_centers = (bins[:-1] + bins[1:]) / 2  
log\_curve\_x = np.exp(bin\_centers)  
  
# Get reference histogram counts to scale properly  
linear\_counts, \_ = np.histogram(df\_salary["Average\_Salary"], bins=40)  
scale\_factor = max(linear\_counts) / max(counts)  
log\_curve\_y = counts \* scale\_factor  
  
# --- Add overlay line ---  
fig.add\_trace(go.Scatter(  
 x=log\_curve\_x,  
 y=log\_curve\_y,  
 mode="lines",  
 line=dict(color="#880E4F", width=3), # deeper magenta for contrast  
 name="Log-transformed Overlay"  
))  
  
# --- Keep your layout exactly the same ---  
fig.update\_layout(  
 title=dict(  
 text="<b>Distribution of Average Salaries (with Log Overlay)</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=20)  
 ),  
 xaxis\_title="<b>Average Salary (USD)</b>",  
 yaxis\_title="<b>Number of Job Postings</b>",  
 template="plotly\_white",  
 width=900,  
 height=550,  
 bargap=0.05,  
 font=dict(size=14),  
 margin=dict(l=60, r=40, t=80, b=60),  
 legend=dict(title="Legend", orientation="h", y=-0.2, x=0.3)  
)  
  
fig.update\_xaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
fig.update\_yaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Salary Distribution

The salary distribution is right-skewed, indicating that most job postings fall within the $80K–$150K range. A smaller number of positions extend above $200K, reflecting higher-paying senior and Machine Learning roles. This pattern aligns with national data showing that advanced technical and AI-related positions command higher compensation due to specialized skill requirements (Bureau of Labor Statistics (2024); Bone, Ehlinger, and Stephany (2025)). Such variation underscores the growing salary dispersion across digital and data-driven occupations (Mäkelä and Stephany (2024)).

# Filter out invalid or missing values  
df\_exp = df[(df["Average\_Salary"] > 0) & (df["MIN\_YEARS\_EXPERIENCE"] >= 0)].copy()  
  
# Combine '[None]' and 'Not Remote' into 'Onsite'  
df\_exp["REMOTE\_GROUP"] = df\_exp["REMOTE\_GROUP"].replace({  
 None: "Onsite",  
 "[None]": "Onsite",  
 "Not Remote": "Onsite"  
})  
  
# Add gentle jitter to make overlapping points clearer   
jitter\_strength = 0.2  
df\_exp["MIN\_YEARS\_EXPERIENCE\_JITTER"] = df\_exp["MIN\_YEARS\_EXPERIENCE"] + np.random.uniform(  
 low=-jitter\_strength, high=jitter\_strength, size=len(df\_exp)  
)  
df\_exp["Average\_Salary\_JITTER"] = df\_exp["Average\_Salary"] + np.random.uniform(  
 low=-2000, high=2000, size=len(df\_exp)  
)  
  
# Use a rainbow-like palette manually   
rainbow\_colors = pc.sample\_colorscale("Rainbow", [0.0, 0.25, 0.5, 0.75, 1.0])  
  
# Create scatter plot with rainbow gradient colors   
fig = px.scatter(  
 df\_exp,  
 x="MIN\_YEARS\_EXPERIENCE\_JITTER",  
 y="Average\_Salary\_JITTER",  
 color="REMOTE\_GROUP",  
 trendline="ols",  
 title="Salary vs. Minimum Experience by Remote Type",  
 color\_discrete\_sequence=rainbow\_colors,   
 opacity=0.7,  
 height=550  
)  
  
fig.update\_layout(  
 title=dict(  
 text="<b>Salary vs. Minimum Experience by Remote Type</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=20)  
 ),  
 xaxis\_title="<b>Minimum Years of Experience</b>",  
 yaxis\_title="<b>Average Salary (USD)</b>",  
 template="plotly\_white",  
 width=900,  
 height=550,  
 font=dict(size=14),  
 legend\_title\_text="<b>Remote Work Type</b>",  
 margin=dict(l=60, r=40, t=80, b=60)  
)  
  
# Final touch: consistent visual polish  
fig.update\_traces(marker=dict(size=6))  
fig.update\_xaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
fig.update\_yaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Salary vs Experience

The scatterplot shows a clear positive relationship between experience and salary across all remote work types. As minimum years of experience increase, average salary consistently rises, supporting labor economics findings that work experience contributes directly to wage growth (Mincer (1974); Bureau of Labor Statistics (2024)). Among the three groups, Remote and Hybrid Remote roles generally track above Not Remote positions, suggesting that employers may offer higher pay for flexible or remote-friendly jobs (McKinsey & Company (2022)). This mirrors recent research showing that post-pandemic flexibility often correlates with higher total compensation, especially for data professionals (Glassdoor Economic Research (2024)).

# Filter valid salary data  
df\_roles = df[df["Average\_Salary"] > 0].copy()  
  
# Use a solid rainbow palette  
rainbow\_colors = pc.sample\_colorscale("Rainbow", [0.0, 0.2, 0.4, 0.6, 0.8, 1.0])  
  
# Create the boxplot  
fig = px.box(  
 df\_roles,  
 x="ROLE\_GROUP",  
 y="Average\_Salary",  
 color="ROLE\_GROUP",  
 color\_discrete\_sequence=rainbow\_colors,   
 title="Salary Comparison Across Role Categories",  
 points="outliers", # show only outliers for clarity  
 height=550  
)  
  
# Layout polish  
fig.update\_layout(  
 title=dict(  
 text="<b>Salary Comparison Across Role Categories</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=20)  
 ),  
 xaxis\_title="<b>Role Group</b>",  
 yaxis\_title="<b>Average Salary (USD)</b>",  
 template="plotly\_white",  
 width=900,  
 height=550,  
 font=dict(size=14),  
 legend\_title\_text="<b>ROLE\_GROUP</b>",  
 margin=dict(l=60, r=40, t=80, b=120)  
)  
  
# Axis & tick styling  
fig.update\_xaxes(  
 tickangle=30,  
 tickfont=dict(size=12),  
 showgrid=False  
)  
fig.update\_yaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Role-Based Salary Trends

The boxplot shows clear salary variation across professional role groups. Network and Systems Engineering roles offer the highest median salaries and the widest overall pay range, consistent with labor market reports highlighting elevated compensation for infrastructure and AI-related talent (Lightcast (2024)). In contrast, Business Analysis and Marketing Specialist roles cluster around lower medians, reflecting standardized pay structures common in business-facing positions (Glassdoor Economic Research (2024)). Overall, the trend highlights that technical specialization and data infrastructure expertise command higher pay within analytics fields (Bone, Ehlinger, and Stephany (2025)).

# Copy and clean up data  
df\_remote = df[df["Average\_Salary"] > 0].copy()  
  
# Combine '[None]' and 'Not Remote' into 'Onsite'  
df\_remote["REMOTE\_GROUP"] = df\_remote["REMOTE\_GROUP"].replace({  
 None: "Onsite",  
 "[None]": "Onsite",  
 "Not Remote": "Onsite"  
})  
  
# Define custom order for clarity  
remote\_order = ["Onsite", "Hybrid Remote", "Remote"]  
  
# Use a solid rainbow palette  
rainbow\_colors = pc.sample\_colorscale("Rainbow", [0.0, 0.5, 1.0])  
  
# Build polished boxplot  
fig = px.box(  
 df\_remote,  
 x="REMOTE\_GROUP",  
 y="Average\_Salary",  
 color="REMOTE\_GROUP",  
 category\_orders={"REMOTE\_GROUP": remote\_order},  
 color\_discrete\_sequence=rainbow\_colors,  
 title="Salary Distribution by Remote Work Type",  
 points="outliers", # show only outliers  
 height=550  
)  
  
# Aesthetic refinements  
fig.update\_layout(  
 title=dict(  
 text="<b>Salary Distribution by Remote Work Type</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=20)  
 ),  
 xaxis\_title="<b>Remote Work Type</b>",  
 yaxis\_title="<b>Average Salary (USD)</b>",  
 template="plotly\_white",  
 width=900,  
 height=550,  
 font=dict(size=14),  
 legend\_title\_text="<b>Remote Work Type</b>",  
 legend=dict(  
 orientation="v",  
 yanchor="top",  
 y=0.98,  
 xanchor="right",  
 x=1.12,  
 font=dict(size=12)  
 ),  
 margin=dict(l=60, r=120, t=80, b=100)  
)  
  
# Axis polish  
fig.update\_xaxes(  
 tickangle=0,  
 showgrid=False,  
 tickfont=dict(size=13)  
)  
fig.update\_yaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Remote Work vs Salary Trends

The salary distribution varies noticeably by remote work type. Remote roles show the highest median salaries and a slightly wider interquartile range, indicating greater earning potential and variability among remote positions. Hybrid Remote jobs follow closely, while Onsite roles exhibit lower medians and tighter spreads. This mirrors workforce evidence that remote-capable roles—particularly in analytics and technology—carry a wage premium and higher demand (McKinsey & Company (2022); Lightcast (2024)). The findings align with broader trends emphasizing flexibility and digital collaboration as valued aspects of compensation (Glassdoor Economic Research (2024)).

# Use the detected skill column  
skills\_column = "COMMON\_SKILLS\_NAME" # replace if another had more data  
  
# Flatten and clean skills  
skills\_flat = [  
 s.strip().replace('"', '') # remove quotation marks  
 for sublist in df[skills\_column].dropna().astype(str).str.split(',')  
 for s in sublist if s.strip()  
]  
  
# Count top 15 skills  
skill\_counts = pd.DataFrame(  
 Counter(skills\_flat).most\_common(15),  
 columns=["Skill", "Count"]  
)  
  
# Generate a rainbow color palette for the bars  
rainbow\_colors = pc.sample\_colorscale("Rainbow", [i / 14 for i in range(15)])  
  
# Bar chart  
fig = px.bar(  
 skill\_counts,  
 x="Skill",  
 y="Count",  
 title="Top 15 Most Frequent Skills",  
 color="Skill",  
 color\_discrete\_sequence=rainbow\_colors  
)  
  
# Clean layout  
fig.update\_layout(  
 title=dict(  
 text="<b>Top 15 Most Frequent Skills</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=22)  
 ),  
 xaxis\_title="<b>Skill</b>",  
 yaxis\_title="<b>Number of Job Postings</b>",  
 template="plotly\_white",  
 width=900,  
 height=550,  
 font=dict(size=14),  
 margin=dict(l=60, r=40, t=80, b=120)  
)  
  
# Axis styling  
fig.update\_xaxes(  
 tickangle=35,  
 tickfont=dict(size=12),  
 showgrid=False,  
 tickvals=list(range(len(skill\_counts))),  
 ticktext=[skill.replace(" (Programming Language)", "") for skill in skill\_counts["Skill"]]  
)  
fig.update\_yaxes(showgrid=True, gridcolor="lightgray", zeroline=False)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Top Skills Frequency

The top skills highlight a blend of technical, analytical, and interpersonal competencies valued across data-driven roles. Communication and Data Analysis appear most frequently, underscoring the importance of translating insights into business impact. SQL, Python, and Microsoft Excel remain core technical skills, while Project Management and Leadership emphasize strategic coordination (DataCamp (2025); Lightcast (2024)). This distribution reflects the “T-shaped” professional model, where deep technical knowledge is complemented by broad problem-solving and collaboration abilities (Bone, Ehlinger, and Stephany (2025); Mäkelä and Stephany (2024)).

# Compute correlation matrix  
corr\_cols = ["Average\_Salary", "MIN\_YEARS\_EXPERIENCE", "MAX\_YEARS\_EXPERIENCE"]  
corr = df[corr\_cols].corr().round(2)  
  
# Create a rainbow colorscale  
rainbow\_colorscale = pc.make\_colorscale(pc.sample\_colorscale("Rainbow", [0.0, 0.25, 0.5, 0.75, 1.0]))  
  
# Create heatmap  
fig = ff.create\_annotated\_heatmap(  
 z=corr.values,  
 x=corr.columns.tolist(),  
 y=corr.columns.tolist(),  
 colorscale=rainbow\_colorscale,  
 showscale=True,  
 annotation\_text=corr.values,  
 font\_colors=["black"],  
 hoverinfo="none"  
)  
  
# Layout adjustments  
fig.update\_layout(  
 title=dict(  
 text="<b>Feature Correlation Matrix</b>",  
 x=0.5,  
 xanchor="center",  
 font=dict(size=20)  
 ),  
 template="plotly\_white",  
 width=700,  
 height=500,  
 margin=dict(l=100, r=100, t=120, b=80),   
 font=dict(size=14)  
)  
  
# Axis polish  
fig.update\_xaxes(  
 side="bottom",  
 tickangle=35,  
 tickfont=dict(size=12),  
 title\_standoff=10  
)  
fig.update\_yaxes(  
 tickfont=dict(size=12),  
 autorange="reversed"  
)  
  
fig.show()

Unable to display output for mime type(s): text/html

### Feature Correlation Analysis Heatmap

This correlation analysis evaluates the linear relationships between Average Salary, Minimum Years of Experience, and Maximum Years of Experience. The results show that salary is moderately correlated with both experience metrics — 0.51 with minimum and 0.58 with maximum years of experience. This indicates that as experience increases, compensation tends to rise, though not perfectly linearly (Bureau of Labor Statistics (2024); Mincer (1974)). The strong correlation (1.0) between minimum and maximum experience suggests these variables are closely tied — employers typically define experience ranges that scale together (Lightcast (2024)).

Overall, the heatmap confirms that experience is a meaningful predictor of salary, supporting its inclusion as a key numerical feature in subsequent regression modeling (Glassdoor Economic Research (2024); McKinsey & Company (2022)).

Bone, M., E. G. Ehlinger, and F. Stephany. (2025): “Skills or degree? The rise of skill-based hiring for AI and green jobs,” *Technological Forecasting and Social Change*, 214, 124042.

Bureau of Labor Statistics. (2024): “Occupational Employment and Wage Statistics,”<https://www.bls.gov/oes/>.

DataCamp. (2025): “Machine Learning Courses | DataCamp,”

Glassdoor Economic Research. (2024): “Data Science Salaries and Market Trends,”

Lightcast. (2024): “Global Labor Market Insights Report,”<https://lightcast.io/resources>.

Mäkelä, E., and F. Stephany. (2024): “Complement or substitute? How AI increases the demand for human skills,” *arXiv preprint arXiv:2412.19754*,.

McKinsey & Company. (2022): The State of Remote Work: Productivity, Pay, and Preferences,McKinsey Global Institute.

Mincer, J. (1974): *Schooling, Experience, and Earnings*, National Bureau of Economic Research.