\*— title: Assignment 04 author: - name: Emily Sundberg affiliations: - id: bu name: Boston University city: Boston state: MA number-sections: true date: ‘2025-10-08’ date-modified: today date-format: long format: html: theme: cerulean toc: true toc-depth: 2

execute: echo: true eval: true output: true freeze: auto —

from pyspark.sql import SparkSession  
import pandas as pd  
import plotly.express as px  
import plotly.io as pio  
import numpy as np  
  
np.random.seed(42)  
  
pio.renderers.default = "notebook+notebook\_connected+vscode"  
  
# Initialize Spark Session  
spark = SparkSession.builder.appName("LightcastData").getOrCreate()  
  
# Load Data  
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","true").option("escape", "\"").csv("./data/lightcast\_job\_postings.csv")  
  
# Show Schema and Sample Data  
#print("---This is Diagnostic check, No need to print it in the final doc---")  
  
# df.printSchema() # comment this line when rendering the submission  
df.show(5)

[Stage 226:> (0 + 1) / 1]

+--------------------+-----------------+----------------------+----------+--------+---------+--------+--------------------+--------------------+--------------------+-----------+-------------------+--------------------+--------------------+---------------+----------------+--------+--------------------+-----------+-------------------+----------------+---------------------+-------------+-------------------+-------------+------------------+---------------+--------------------+--------------------+--------------------+-------------+------+-----------+----------------+-------------------+---------+-----------+--------------------+--------------------+-------------+------+--------------+-----+--------------------+-----+----------+---------------+--------------------+---------------+--------------------+------------+--------------------+------------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------------------+-------------------+--------------------+--------------------+--------------------+--------------------+-----------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+----------+---------------+----------+---------------+---------------+--------------------+--------------+--------------------+--------------------------+-------------------------------+--------------------+-------------------------+-----------------------------+----------------------------------+-----------------+----------------------+-----------------------+----------------------------+------------------+-----------------------+-------+--------------------+-------+--------------------+-------+---------------+-------+---------------+-----------------+----------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+  
| 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|  
+--------------------+-----------------+----------------------+----------+--------+---------+--------+--------------------+--------------------+--------------------+-----------+-------------------+--------------------+--------------------+---------------+----------------+--------+--------------------+-----------+-------------------+----------------+---------------------+-------------+-------------------+-------------+------------------+---------------+--------------------+--------------------+--------------------+-------------+------+-----------+----------------+-------------------+---------+-----------+--------------------+--------------------+-------------+------+--------------+-----+--------------------+-----+----------+---------------+--------------------+---------------+--------------------+------------+--------------------+------------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------------------+-------------------+--------------------+--------------------+--------------------+--------------------+-----------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+----------+---------------+----------+---------------+---------------+--------------------+--------------+--------------------+--------------------------+-------------------------------+--------------------+-------------------------+-----------------------------+----------------------------------+-----------------+----------------------+-----------------------+----------------------------+------------------+-----------------------+-------+--------------------+-------+--------------------+-------+---------------+-------+---------------+-----------------+----------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+  
|1f57d95acf4dc67ed...| 9/6/2024| 2024-09-06 20:32:...| 0|6/2/2024| 6/8/2024| 6| [\n "Company"\n]|[\n "brassring.c...|[\n "https://sjo...| []| NULL|Enterprise Analys...|31-May-2024\n\nEn...| 6/8/2024| 6| 894731| Murphy USA| Murphy USA| false| [\n 2\n]| [\n "Bachelor's ...| 2| Bachelor's degree| NULL| NULL| 1|Full-time (> 32 h...| 2| 2| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 33.20...|RWwgRG9yYWRvLCBBUg==|El Dorado, AR| 5139| Union, AR|20980| El Dorado, AR| 5| Arkansas| 5139| Union, AR| 5139| Union, AR| 20980| El Dorado, AR| 20980| El Dorado, AR| 44| Retail Trade| 441|Motor Vehicle and...| 4413|Automotive Parts,...| 44133|Automotive Parts ...|441330|Automotive Parts ...|ET29C073C03D1F86B4|Enterprise Analysts|enterprise analys...|[\n "KS126DB6T06...|[\n "Merchandisi...|[\n "KS126DB6T06...| [\n "Merchandisi...| []| []|[\n "KS126706DPF...|[\n "Mathematics...|[\n "KS440W865GC...|[\n "SQL (Progra...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...|[\n "45.0601",\n...|[\n "Economics, ...|[\n "45.06",\n ...|[\n "Economics",...|[\n "45",\n "27...|[\n "Social Scie...| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101011| General ERP Analy...| 2310| Business Intellig...| 23101011| General ERP Analy...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| [\n 7\n]| [\n "Artificial ...| 44| Retail Trade| 441|Motor Vehicle and...| 4413|Automotive Parts,...| 44133|Automotive Parts ...| 441330|Automotive Parts ...|  
|0cb072af26757b6c4...| 8/2/2024| 2024-08-02 17:08:...| 0|6/2/2024| 8/1/2024| NULL| [\n "Job Board"\n]| [\n "maine.gov"\n]|[\n "https://job...| []| NULL|Oracle Consultant...|Oracle Consultant...| 8/1/2024| NULL| 133098|Smx Corporation L...| SMX| true| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 1|Full-time (> 32 h...| 3| 3| false| NULL| 1| Remote| NULL| NULL| NULL|{\n "lat": 44.31...| QXVndXN0YSwgTUU=| Augusta, ME| 23011| Kennebec, ME|12300|Augusta-Watervill...| 23| Maine| 23011| Kennebec, ME| 23011| Kennebec, ME| 12300|Augusta-Watervill...| 12300|Augusta-Watervill...| 56|Administrative an...| 561|Administrative an...| 5613| Employment Services| 56132|Temporary Help Se...|561320|Temporary Help Se...|ET21DDA63780A7DC09| Oracle Consultants|oracle consultant...|[\n "KS122626T55...|[\n "Procurement...|[\n "KS122626T55...| [\n "Procurement...| []| []| []| []|[\n "BGSBF3F508F...|[\n "Oracle Busi...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101012| Oracle Consultant...| 2310| Business Intellig...| 23101012| Oracle Consultant...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 56|Administrative an...| 561|Administrative an...| 5613| Employment Services| 56132|Temporary Help Se...| 561320|Temporary Help Se...|  
|85318b12b3331fa49...| 9/6/2024| 2024-09-06 20:32:...| 1|6/2/2024| 7/7/2024| 35| [\n "Job Board"\n]|[\n "dejobs.org"\n]|[\n "https://dej...| []| NULL| Data Analyst|Taking care of pe...| 6/10/2024| 8|39063746| Sedgwick| Sedgwick| false| [\n 2\n]| [\n "Bachelor's ...| 2| Bachelor's degree| NULL| NULL| 1|Full-time (> 32 h...| 5| NULL| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 32.77...| RGFsbGFzLCBUWA==| Dallas, TX| 48113| Dallas, TX|19100|Dallas-Fort Worth...| 48| Texas| 48113| Dallas, TX| 48113| Dallas, TX| 19100|Dallas-Fort Worth...| 19100|Dallas-Fort Worth...| 52|Finance and Insur...| 524|Insurance Carrier...| 5242|Agencies, Brokera...| 52429|Other Insurance R...|524291| Claims Adjusting|ET3037E0C947A02404| Data Analysts| data analyst|[\n "KS1218W78FG...|[\n "Management"...|[\n "ESF3939CE1F...| [\n "Exception R...|[\n "KS683TN76T7...|[\n "Security Cl...|[\n "KS1218W78FG...|[\n "Management"...|[\n "KS126HY6YLT...|[\n "Microsoft O...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231113|Data / Data Minin...| 23111310| Data Analyst| 2311| Data Analysis and...| 23111310| Data Analyst| 231113| Data / Data Minin...| 2311| Data Analysis and...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 52|Finance and Insur...| 524|Insurance Carrier...| 5242|Agencies, Brokera...| 52429|Other Insurance R...| 524291| Claims Adjusting|  
|1b5c3941e54a1889e...| 9/6/2024| 2024-09-06 20:32:...| 1|6/2/2024|7/20/2024| 48| [\n "Job Board"\n]|[\n "disabledper...|[\n "https://www...| []| NULL|Sr. Lead Data Mgm...|About this role:\...| 6/12/2024| 10|37615159| Wells Fargo|Wells Fargo| false| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 1|Full-time (> 32 h...| 3| NULL| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 33.44...| UGhvZW5peCwgQVo=| Phoenix, AZ| 4013| Maricopa, AZ|38060|Phoenix-Mesa-Chan...| 4| Arizona| 4013| Maricopa, AZ| 4013| Maricopa, AZ| 38060|Phoenix-Mesa-Chan...| 38060|Phoenix-Mesa-Chan...| 52|Finance and Insur...| 522|Credit Intermedia...| 5221|Depository Credit...| 52211| Commercial Banking|522110| Commercial Banking|ET2114E0404BA30075|Management Analysts|sr lead data mgmt...|[\n "KS123QX62QY...|[\n "Exit Strate...|[\n "KS123QX62QY...| [\n "Exit Strate...| []| []|[\n "KS7G6NP6R6L...|[\n "Reliability...|[\n "KS4409D76NW...|[\n "SAS (Softwa...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231113|Data / Data Minin...| 23111310| Data Analyst| 2311| Data Analysis and...| 23111310| Data Analyst| 231113| Data / Data Minin...| 2311| Data Analysis and...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| [\n 6\n]| [\n "Data Privac...| 52|Finance and Insur...| 522|Credit Intermedia...| 5221|Depository Credit...| 52211| Commercial Banking| 522110| Commercial Banking|  
|cb5ca25f02bdf25c1...| 6/19/2024| 2024-06-19 07:00:00| 0|6/2/2024|6/17/2024| 15|[\n "FreeJobBoar...|[\n "craigslist....|[\n "https://mod...| []| NULL|Comisiones de $10...|Comisiones de $10...| 6/17/2024| 15| 0| Unclassified| LH/GM| false| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 3|Part-time / full-...| NULL| NULL| false| 92500| 0| [None]| year| 150000| 35000|{\n "lat": 37.63...| TW9kZXN0bywgQ0E=| Modesto, CA| 6099|Stanislaus, CA|33700| Modesto, CA| 6|California| 6099| Stanislaus, CA| 6099| Stanislaus, CA| 33700| Modesto, CA| 33700| Modesto, CA| 99|Unclassified Indu...| 999|Unclassified Indu...| 9999|Unclassified Indu...| 99999|Unclassified Indu...|999999|Unclassified Indu...|ET0000000000000000| Unclassified|comisiones de por...| []| []| []| []| []| []| []| []| []| []|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101012| Oracle Consultant...| 2310| Business Intellig...| 23101012| Oracle Consultant...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 99|Unclassified Indu...| 999|Unclassified Indu...| 9999|Unclassified Indu...| 99999|Unclassified Indu...| 999999|Unclassified Indu...|  
+--------------------+-----------------+----------------------+----------+--------+---------+--------+--------------------+--------------------+--------------------+-----------+-------------------+--------------------+--------------------+---------------+----------------+--------+--------------------+-----------+-------------------+----------------+---------------------+-------------+-------------------+-------------+------------------+---------------+--------------------+--------------------+--------------------+-------------+------+-----------+----------------+-------------------+---------+-----------+--------------------+--------------------+-------------+------+--------------+-----+--------------------+-----+----------+---------------+--------------------+---------------+--------------------+------------+--------------------+------------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------------------+-------------------+--------------------+--------------------+--------------------+--------------------+-----------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+----------+---------------+----------+---------------+---------------+--------------------+--------------+--------------------+--------------------------+-------------------------------+--------------------+-------------------------+-----------------------------+----------------------------------+-----------------+----------------------+-----------------------+----------------------------+------------------+-----------------------+-------+--------------------+-------+--------------------+-------+---------------+-------+---------------+-----------------+----------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+  
only showing top 5 rows

# create eda data frame

from pyspark.sql.functions import col, pow  
from pyspark.ml.feature import StringIndexer, OneHotEncoder, VectorAssembler  
from pyspark.ml import Pipeline  
  
eda\_cols = [  
 "SALARY",  
 "MIN\_YEARS\_EXPERIENCE",  
 "DURATION",  
 "COMPANY\_IS\_STAFFING",  
 "IS\_INTERNSHIP",  
 "STATE\_NAME",  
 "REMOTE\_TYPE\_NAME",  
 "EMPLOYMENT\_TYPE\_NAME",  
 "MIN\_EDULEVELS\_NAME"  
]  
  
df\_eda = df.select(eda\_cols)  
df\_eda.show(5)

+------+--------------------+--------+-------------------+-------------+----------+----------------+--------------------+-------------------+  
|SALARY|MIN\_YEARS\_EXPERIENCE|DURATION|COMPANY\_IS\_STAFFING|IS\_INTERNSHIP|STATE\_NAME|REMOTE\_TYPE\_NAME|EMPLOYMENT\_TYPE\_NAME| MIN\_EDULEVELS\_NAME|  
+------+--------------------+--------+-------------------+-------------+----------+----------------+--------------------+-------------------+  
| NULL| 2| 6| false| false| Arkansas| [None]|Full-time (> 32 h...| Bachelor's degree|  
| NULL| 3| NULL| true| false| Maine| Remote|Full-time (> 32 h...|No Education Listed|  
| NULL| 5| 35| false| false| Texas| [None]|Full-time (> 32 h...| Bachelor's degree|  
| NULL| 3| 48| false| false| Arizona| [None]|Full-time (> 32 h...|No Education Listed|  
| 92500| NULL| 15| false| false|California| [None]|Part-time / full-...|No Education Listed|  
+------+--------------------+--------+-------------------+-------------+----------+----------------+--------------------+-------------------+  
only showing top 5 rows

# Dealing With Missing Data

## Analyzing the Missing Data

### Bar Chart

from pyspark.sql.functions import col, sum as spark\_sum, when, trim, length  
import hvplot.pandas  
  
missing\_df = df\_eda.select([  
 spark\_sum(  
 when(col(c).isNull() | (length(trim(col(c)))== 0),1).otherwise(0)).alias(c)  
 for c in df\_eda.columns  
])  
  
missing\_pd = missing\_df.toPandas().T.reset\_index()  
missing\_pd.columns = ["column", "missing\_count"]  
  
total\_rows = df\_eda.count()  
  
missing\_pd["missing\_percent"]=100\*missing\_pd["missing\_count"]/total\_rows  
  
missing\_pd.sort\_values("missing\_percent", ascending=False).hvplot.bar(  
 x="column",  
 y="missing\_percent",  
 rot=90,  
 title="Percentage of Missing Values by Column",  
 height = 600,  
 width = 900,  
 ylabel = "Missing Percentage (%)",  
 xlabel = "Features"  
).opts(xrotation=45)  
  
#missing\_pd.head()

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

Unable to display output for mime type(s): application/javascript, application/vnd.holoviews\_load.v0+json

Unable to display output for mime type(s): application/javascript, application/vnd.holoviews\_load.v0+json

Unable to display output for mime type(s): application/vnd.holoviews\_exec.v0+json, text/html

[Stage 229:> (0 + 1) / 1] [Stage 232:> (0 + 1) / 1]

:Bars [column] (missing\_percent)

### Heatmap

import pandas as pd  
  
df\_sample = df\_eda.sample(fraction = .05, seed = 42).toPandas()  
  
missing\_mask = df\_sample.isnull()  
  
missing\_long = (  
 missing\_mask.reset\_index()  
 .melt(id\_vars = "index", var\_name = "column", value\_name = "is\_missing")  
)  
  
missing\_long["is\_missing"] = missing\_long["is\_missing"].astype(int)  
  
missing\_long.hvplot.heatmap(  
 x="column",  
 y="index",  
 C = "is\_missing",  
 cmap = "Blues",  
 width = 900,  
 height = 500,  
 title = "Heatmap of Missing Values (5%)"  
).opts(xrotation=45)

[Stage 235:> (0 + 1) / 1]

:HeatMap [column,index] (is\_missing)

### Unique Values

from pyspark.sql.functions import countDistinct  
  
df\_eda.select([  
 countDistinct(c).alias(c+"\_nunique")  
 for c in df\_eda.columns  
]).show(truncate=False)

[Stage 236:> (0 + 1) / 1]

+--------------+----------------------------+----------------+---------------------------+---------------------+------------------+------------------------+----------------------------+--------------------------+  
|SALARY\_nunique|MIN\_YEARS\_EXPERIENCE\_nunique|DURATION\_nunique|COMPANY\_IS\_STAFFING\_nunique|IS\_INTERNSHIP\_nunique|STATE\_NAME\_nunique|REMOTE\_TYPE\_NAME\_nunique|EMPLOYMENT\_TYPE\_NAME\_nunique|MIN\_EDULEVELS\_NAME\_nunique|  
+--------------+----------------------------+----------------+---------------------------+---------------------+------------------+------------------------+----------------------------+--------------------------+  
|6052 |16 |60 |2 |2 |51 |4 |3 |6 |  
+--------------+----------------------------+----------------+---------------------------+---------------------+------------------+------------------------+----------------------------+--------------------------+

### Replacing Null Values and Errors

# REMOTE TYPE NAME  
df\_eda = df\_eda.withColumn(  
 "REMOTE\_TYPE\_NAME",  
 when(col("REMOTE\_TYPE\_NAME") == "Remote","Remote")  
 .when(col("REMOTE\_TYPE\_NAME") == "[None]","In-Person")  
 .when(col("REMOTE\_TYPE\_NAME") == "Not Remote","In-Person")  
 .when(col("REMOTE\_TYPE\_NAME") == "Hybrid Remote","Hybrid")  
 .when(col("REMOTE\_TYPE\_NAME").isNull(), "In-Person")  
 .otherwise(col("REMOTE\_TYPE\_NAME"))  
)  
  
# EMPLOYMENT TYPE NAME  
  
df\_eda = df\_eda.withColumn(  
 "EMPLOYMENT\_TYPE\_NAME",  
 when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time / full-time","Flexible")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Part-time (â‰¤ 32 hours)","Part-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME") == "Full-time (> 32 hours)","Full-Time")  
 .when(col("EMPLOYMENT\_TYPE\_NAME").isNull(), "Full-Time")  
 .otherwise(col("EMPLOYMENT\_TYPE\_NAME"))  
)  
  
# Minimum Education Levels  
df\_eda = df\_eda.withColumn(  
 "MIN\_EDULEVELS\_NAME",  
 when(col("MIN\_EDULEVELS\_NAME").isNull(), "No Education Listed")  
 .otherwise(col("MIN\_EDULEVELS\_NAME"))  
)  
  
# State Name  
df\_eda = df\_eda.withColumn(  
 "STATE\_NAME",  
 when(col("STATE\_NAME").isNull(), "Unknown")  
 .otherwise(col("STATE\_NAME"))  
)  
  
# Company Staffing  
df\_eda = df\_eda.withColumn(  
 "COMPANY\_IS\_STAFFING",  
 when(col("COMPANY\_IS\_STAFFING").isNull(), False)  
 .otherwise(col("COMPANY\_IS\_STAFFING"))  
)  
  
# Internship  
df\_eda = df\_eda.withColumn(  
 "IS\_INTERNSHIP",  
 when(col("IS\_INTERNSHIP").isNull(), False)  
 .otherwise(col("IS\_INTERNSHIP"))  
)  
  
categorical\_cols = [  
 "STATE\_NAME",  
 "REMOTE\_TYPE\_NAME",  
 "EMPLOYMENT\_TYPE\_NAME",  
 "MIN\_EDULEVELS\_NAME",  
 "COMPANY\_IS\_STAFFING",  
 "IS\_INTERNSHIP"  
]  
  
for colname in categorical\_cols:  
 print(f"\n----{colname} ----")  
 df\_eda.select(colname).distinct().show(60,truncate = False)

----STATE\_NAME ----

[Stage 242:> (0 + 1) / 1]

+---------------------------------------+  
|STATE\_NAME |  
+---------------------------------------+  
|Utah |  
|Hawaii |  
|Minnesota |  
|Ohio |  
|Arkansas |  
|Oregon |  
|Texas |  
|North Dakota |  
|Pennsylvania |  
|Connecticut |  
|Nebraska |  
|Vermont |  
|Nevada |  
|Washington |  
|Illinois |  
|Oklahoma |  
|Unknown |  
|Delaware |  
|Alaska |  
|New Mexico |  
|West Virginia |  
|Missouri |  
|Rhode Island |  
|Georgia |  
|Montana |  
|Michigan |  
|Virginia |  
|Washington, D.C. (District of Columbia)|  
|North Carolina |  
|Wyoming |  
|Kansas |  
|New Jersey |  
|Maryland |  
|Alabama |  
|Arizona |  
|Iowa |  
|Massachusetts |  
|Kentucky |  
|Louisiana |  
|Mississippi |  
|Tennessee |  
|New Hampshire |  
|Indiana |  
|Florida |  
|Idaho |  
|South Carolina |  
|South Dakota |  
|California |  
|New York |  
|Wisconsin |  
|Maine |  
|Colorado |  
+---------------------------------------+  
  
  
----REMOTE\_TYPE\_NAME ----

[Stage 245:> (0 + 1) / 1]

+----------------+  
|REMOTE\_TYPE\_NAME|  
+----------------+  
|Remote |  
|Hybrid |  
|In-Person |  
+----------------+  
  
  
----EMPLOYMENT\_TYPE\_NAME ----

[Stage 248:> (0 + 1) / 1]

+--------------------+  
|EMPLOYMENT\_TYPE\_NAME|  
+--------------------+  
|Flexible |  
|Full-Time |  
|Part-Time |  
+--------------------+  
  
  
----MIN\_EDULEVELS\_NAME ----

[Stage 251:> (0 + 1) / 1]

+----------------------------+  
|MIN\_EDULEVELS\_NAME |  
+----------------------------+  
|Bachelor's degree |  
|Ph.D. or professional degree|  
|High school or GED |  
|Master's degree |  
|No Education Listed |  
|Associate degree |  
+----------------------------+  
  
  
----COMPANY\_IS\_STAFFING ----

[Stage 254:> (0 + 1) / 1]

+-------------------+  
|COMPANY\_IS\_STAFFING|  
+-------------------+  
|true |  
|false |  
+-------------------+  
  
  
----IS\_INTERNSHIP ----

[Stage 257:> (0 + 1) / 1]

+-------------+  
|IS\_INTERNSHIP|  
+-------------+  
|true |  
|false |  
+-------------+

# Feature Engineering

from pyspark.sql.functions import col, when, expr  
  
  
med\_duration = df\_eda.approxQuantile("DURATION",[0.5], 0.01)[0]  
  
df\_eda = df\_eda.withColumn(  
 "DURATION",  
 when(col("DURATION").isNull(), med\_duration)  
)  
  
overall\_med\_sal = df\_eda.approxQuantile("SALARY", [0.5],0.01)[0]  
  
#print(overall\_med\_sal)  
  
median\_sal\_employment\_type = df\_eda.groupBy("EMPLOYMENT\_TYPE\_NAME").agg(expr("percentile\_approx(SALARY,0.5)").alias("median\_sal\_employment\_type"))  
  
df\_eda = df\_eda.join(median\_sal\_employment\_type, on="EMPLOYMENT\_TYPE\_NAME", how="left")  
  
df\_eda = df\_eda.withColumn("SALARY", when(col("SALARY").isNull(),  
 when(col("median\_sal\_employment\_type").isNotNull(), col("median\_sal\_employment\_type"))  
 .otherwise(overall\_med\_sal)).otherwise(col("SALARY")))

[Stage 260:> (0 + 1) / 1] [Stage 261:> (0 + 1) / 1]

df\_feature = df\_eda.dropna(subset=[  
 "SALARY","MIN\_YEARS\_EXPERIENCE","MIN\_EDULEVELS\_NAME", "STATE\_NAME","EMPLOYMENT\_TYPE\_NAME","REMOTE\_TYPE\_NAME","IS\_INTERNSHIP","COMPANY\_IS\_STAFFING", "DURATION"  
])  
  
categorical\_cols = ["MIN\_EDULEVELS\_NAME", "EMPLOYMENT\_TYPE\_NAME", "REMOTE\_TYPE\_NAME","STATE\_NAME"]  
  
indexers = [StringIndexer(inputCol=col, outputCol = f"{col}\_idx", handleInvalid = 'skip')for col in categorical\_cols]  
  
encoders = [OneHotEncoder(inputCol=f"{col}\_idx",outputCol = f"{col}\_vec")for col in categorical\_cols]  
  
assembler = VectorAssembler(  
 inputCols = [  
 "MIN\_YEARS\_EXPERIENCE","DURATION","IS\_INTERNSHIP", "COMPANY\_IS\_STAFFING"  
 ] + [f"{col}\_vec" for col in categorical\_cols],  
 outputCol= "features"  
)  
  
pipeline = Pipeline(stages = indexers + encoders + [assembler])  
data = pipeline.fit(df\_feature).transform(df\_feature)  
  
data = data.withColumn("MIN\_YEARS\_EXPERIENCE\_SQ", pow(col("MIN\_YEARS\_EXPERIENCE"),2))  
  
  
  
  
data.select("SALARY","features").show(5, truncate = False)

[Stage 262:> (0 + 1) / 1][Stage 262:> (0 + 1) / 1][Stage 263:> (0 + 1) / 1][Stage 262:> (0 + 1) / 1] [Stage 271:> (0 + 1) / 1][Stage 272:> (0 + 1) / 1][Stage 271:> (0 + 1) / 1] [Stage 280:> (0 + 1) / 1][Stage 280:> (0 + 1) / 1][Stage 281:> (0 + 1) / 1] [Stage 289:> (0 + 1) / 1][Stage 289:> (0 + 1) / 1][Stage 290:> (0 + 1) / 1][Stage 289:> (0 + 1) / 1] [Stage 298:> (0 + 1) / 1][Stage 299:> (0 + 1) / 1][Stage 298:> (0 + 1) / 1]

+--------+------------------------------------------+  
|SALARY |features |  
+--------+------------------------------------------+  
|222000.0|(63,[0,1,4,11,24],[10.0,18.0,1.0,1.0,1.0])|  
|101798.0|(63,[0,1,4,11,17],[3.0,18.0,1.0,1.0,1.0]) |  
|98700.0 |(63,[0,1,4,11,14],[2.0,18.0,1.0,1.0,1.0]) |  
|101550.0|(63,[0,1,4,12,14],[2.0,18.0,1.0,1.0,1.0]) |  
|100000.0|(63,[0,1,7,11,18],[1.0,18.0,1.0,1.0,1.0]) |  
+--------+------------------------------------------+  
only showing top 5 rows

# Train/Test Split

regression\_train, regression\_test = data.randomSplit([0.75,0.25],seed=42)  
  
print((data.count(), len(data.columns)))  
print((regression\_train.count(), len(regression\_train.columns)))  
print((regression\_test.count(), len(regression\_test.columns)))

[Stage 304:> (0 + 1) / 1][Stage 305:> (0 + 1) / 1][Stage 304:> (0 + 1) / 1]

(19334, 20)

[Stage 313:> (0 + 1) / 1][Stage 314:> (0 + 1) / 1][Stage 313:> (0 + 1) / 1] [Stage 318:> (0 + 1) / 1]

(14534, 20)

[Stage 322:> (0 + 1) / 1][Stage 323:> (0 + 1) / 1][Stage 322:> (0 + 1) / 1]

(4800, 20)

[Stage 327:> (0 + 1) / 1]

# Linear Regression Model

from pyspark.ml.regression import GeneralizedLinearRegression  
  
feature\_names = assembler.getInputCols()  
  
glr = GeneralizedLinearRegression(  
 featuresCol="features",  
 labelCol="SALARY",  
 family = "gaussian",  
 link = "identity",  
 maxIter = 10,  
 regParam=0.3  
)  
  
glr\_model = glr.fit(regression\_train)  
summary = glr\_model.summary  
  
print("\n --- Regression Summary ---")  
print("Coefficient Standard Errors:", [f"{val:.4f}" for val in summary.coefficientStandardErrors])  
print("T-Values:", [f"{val:.4f}" for val in summary.tValues])  
print("P-Values:",[f"{val:.4f}" for val in summary.pValues])  
  
  
print(f"\nDispersion: {summary.dispersion:.4f}")  
print(f"Null Deviance: {summary.nullDeviance:.4f}")  
print(f"Residual DF Null: {summary.residualDegreeOfFreedomNull}")  
print(f"Deviance: {summary.deviance:.4f}")  
print(f"Residual DF: {summary.residualDegreeOfFreedom}")  
print(f"AIC: {summary.aic:.4f}")

[Stage 331:> (0 + 1) / 1][Stage 332:> (0 + 1) / 1] [Stage 335:> (0 + 1) / 1][Stage 336:> (0 + 1) / 1][Stage 335:> (0 + 1) / 1] [Stage 340:> (0 + 1) / 1]

--- Regression Summary ---

[Stage 341:> (0 + 1) / 1][Stage 342:> (0 + 1) / 1] [Stage 346:> (0 + 1) / 1]

Coefficient Standard Errors: ['75.3627', 'inf', '3299.2815', '893.0967', '9129.4900', '9147.3984', '9183.8577', '9186.5328', '9231.8700', '1808.4642', '2323.5808', '1339.0523', '1436.1112', '7469.8953', '7472.2938', '7487.3352', '7504.5113', '7504.4248', '7508.6605', '7528.8183', '7539.7113', '7543.0325', '7542.7247', '7548.7186', '7553.7983', '7593.7045', '7590.8310', '7616.5959', '7615.0040', '7617.2648', '7622.3009', '7620.5336', '7628.1773', '7659.8343', '7682.2817', '7697.0545', '7720.2852', '7742.4696', '7751.9989', '7831.9358', '7843.7654', '7861.8520', '7859.4928', '7933.5096', '7934.3170', '7931.3088', '8117.7557', '8042.6248', '8071.2966', '8016.3472', '8085.7212', '8108.9893', '8143.6584', '8117.4254', '8093.2312', '8313.0622', '8298.6706', '8389.4048', '8357.1194', '8587.6523', '8891.1782', '9062.1224', '9015.2956', '11939.6494']  
T-Values: ['45.6622', '0.0000', '1.9675', '-3.8479', '-2.0560', '-1.8969', '-3.0329', '-3.9961', '0.0249', '3.4711', '-5.7375', '2.0973', '4.0530', '0.7105', '1.9916', '1.2459', '1.2107', '1.0276', '0.7381', '0.6210', '1.3079', '0.5746', '0.8562', '0.9372', '0.6862', '0.2719', '1.6926', '0.9315', '0.3212', '1.0408', '0.2806', '0.6492', '1.5386', '0.7346', '0.7588', '2.2254', '0.9126', '0.7903', '0.7282', '0.4389', '1.7500', '0.8086', '0.6479', '0.7742', '0.4836', '0.8136', '0.5981', '1.6172', '1.1929', '1.2825', '0.5391', '-0.2562', '1.1190', '0.6921', '0.9614', '0.3243', '0.7181', '0.3635', '0.6372', '1.2128', '0.5635', '0.4246', '0.8885', '8.7980']  
P-Values: ['0.0000', '1.0000', '0.0491', '0.0001', '0.0398', '0.0579', '0.0024', '0.0001', '0.9801', '0.0005', '0.0000', '0.0360', '0.0001', '0.4774', '0.0464', '0.2128', '0.2260', '0.3041', '0.4605', '0.5346', '0.1909', '0.5656', '0.3919', '0.3487', '0.4926', '0.7857', '0.0906', '0.3516', '0.7481', '0.2980', '0.7790', '0.5162', '0.1239', '0.4626', '0.4480', '0.0261', '0.3614', '0.4294', '0.4665', '0.6607', '0.0801', '0.4188', '0.5171', '0.4389', '0.6287', '0.4159', '0.5498', '0.1059', '0.2329', '0.1997', '0.5899', '0.7978', '0.2632', '0.4889', '0.3364', '0.7457', '0.4727', '0.7162', '0.5240', '0.2252', '0.5731', '0.6712', '0.3743', '0.0000']  
  
Dispersion: 834648362.5995

[Stage 350:> (0 + 1) / 1][Stage 350:> (0 + 1) / 1][Stage 351:> (0 + 1) / 1][Stage 350:> (0 + 1) / 1]

Null Deviance: 14812470258074.5137  
Residual DF Null: 14533  
Deviance: 12077361806815.2754  
Residual DF: 14470

[Stage 356:> (0 + 1) / 1][Stage 357:> (0 + 1) / 1][Stage 356:> (0 + 1) / 1]

AIC: 339876.5652

These values tell us that this model is not a good fit.

import pandas as pd  
from tabulate import tabulate  
import pandas as pd   
from IPython.display import HTML  
  
feature\_names = summary.\_call\_java("featureNames")  
coefs = [glr\_model.intercept] +list(glr\_model.coefficients)  
se = list(summary.coefficientStandardErrors)  
tvals = list(summary.tValues)  
pvals = list(summary.pValues)  
  
print("features", len(feature\_names))  
print("coefs", len(coefs))  
print("se", len(se))  
print("tvals", len(tvals))  
print("pvals", len(pvals))  
  
coef\_table = pd.DataFrame({  
 "Feature": ["Intercept"]+feature\_names,  
 "Estimate": [f"{v:.4f}" if v is not None else None for v in coefs],  
 "Std Error": [f"{v:.4f}" if v is not None else None for v in se],  
 "t-stat": [f"{v:.4f}" if v is not None else None for v in tvals],  
 "P-Value": [f"{v:.4f}" if v is not None else None for v in pvals],  
})  
  
coef\_table.to\_csv("glr\_summary.csv", index=False)  
  
  
print(coef\_table)

features 63  
coefs 64  
se 64  
tvals 64  
pvals 64  
 Feature Estimate Std Error t-stat P-Value  
0 Intercept 105044.9016 75.3627 45.6622 0.0000  
1 MIN\_YEARS\_EXPERIENCE 3441.2235 inf 0.0000 1.0000  
2 DURATION 0.0000 3299.2815 1.9675 0.0491  
3 IS\_INTERNSHIP 6491.3080 893.0967 -3.8479 0.0001  
4 COMPANY\_IS\_STAFFING -3436.5476 9129.4900 -2.0560 0.0398  
.. ... ... ... ... ...  
59 STATE\_NAME\_vec\_Alaska 5324.9531 8587.6523 1.2128 0.2252  
60 STATE\_NAME\_vec\_Vermont 10414.9407 8891.1782 0.5635 0.5731  
61 STATE\_NAME\_vec\_Montana 5010.1003 9062.1224 0.4246 0.6712  
62 STATE\_NAME\_vec\_West Virginia 3847.3576 9015.2956 0.8885 0.3743  
63 STATE\_NAME\_vec\_North Dakota 8009.6895 11939.6494 8.7980 0.0000  
  
[64 rows x 5 columns]

# Polynomial Regression Model

poly\_data = data.withColumn("MIN\_YEARS\_EXPERIENCE\_SQ", pow(col("MIN\_YEARS\_EXPERIENCE"),2))  
  
  
assembler\_poly = VectorAssembler(  
 inputCols=[  
 "MIN\_YEARS\_EXPERIENCE",  
 "MIN\_YEARS\_EXPERIENCE\_SQ",  
 "DURATION",  
 "IS\_INTERNSHIP",  
 "COMPANY\_IS\_STAFFING"  
 ]+[f"{col}\_vec" for col in categorical\_cols],  
 outputCol = "poly\_features"  
)  
  
poly\_data = assembler\_poly.transform(poly\_data)  
  
poly\_data.select("SALARY","features","poly\_features").show(5, truncate = False)

[Stage 362:> (0 + 1) / 1][Stage 363:> (0 + 1) / 1][Stage 362:> (0 + 1) / 1]

+--------+------------------------------------------+--------------------------------------------------+  
|SALARY |features |poly\_features |  
+--------+------------------------------------------+--------------------------------------------------+  
|222000.0|(63,[0,1,4,11,24],[10.0,18.0,1.0,1.0,1.0])|(64,[0,1,2,5,12,25],[10.0,100.0,18.0,1.0,1.0,1.0])|  
|101798.0|(63,[0,1,4,11,17],[3.0,18.0,1.0,1.0,1.0]) |(64,[0,1,2,5,12,18],[3.0,9.0,18.0,1.0,1.0,1.0]) |  
|98700.0 |(63,[0,1,4,11,14],[2.0,18.0,1.0,1.0,1.0]) |(64,[0,1,2,5,12,15],[2.0,4.0,18.0,1.0,1.0,1.0]) |  
|101550.0|(63,[0,1,4,12,14],[2.0,18.0,1.0,1.0,1.0]) |(64,[0,1,2,5,13,15],[2.0,4.0,18.0,1.0,1.0,1.0]) |  
|100000.0|(63,[0,1,7,11,18],[1.0,18.0,1.0,1.0,1.0]) |(64,[0,1,2,8,12,19],[1.0,1.0,18.0,1.0,1.0,1.0]) |  
+--------+------------------------------------------+--------------------------------------------------+  
only showing top 5 rows

## Split Data

regression\_train, regression\_test = poly\_data.randomSplit([0.75,0.25], seed=42)  
print((poly\_data.count(), len(poly\_data.columns)))  
print((regression\_train.count(), len(regression\_train.columns)))  
print((regression\_test.count(), len(regression\_test.columns)))

[Stage 368:> (0 + 1) / 1][Stage 369:> (0 + 1) / 1]

(19334, 21)

[Stage 377:> (0 + 1) / 1][Stage 377:> (0 + 1) / 1][Stage 378:> (0 + 1) / 1] [Stage 382:> (0 + 1) / 1]

(14534, 21)

[Stage 386:> (0 + 1) / 1][Stage 387:> (0 + 1) / 1][Stage 386:> (0 + 1) / 1][Stage 387:=================(1 + 0) / 1]

(4800, 21)

[Stage 391:> (0 + 1) / 1]

## regression model

from pyspark.ml.regression import GeneralizedLinearRegression  
  
feature\_names = assembler.getInputCols()  
  
poly\_glr = GeneralizedLinearRegression(  
 featuresCol="features",  
 labelCol="SALARY",  
 family = "gaussian",  
 link = "identity",  
 maxIter = 10,  
 regParam=0.3  
)  
  
poly\_glr\_model = poly\_glr.fit(regression\_train)  
summary = poly\_glr\_model.summary  
  
print("\n --- Regression Summary ---")  
print("Coefficient Standard Errors:", [f"{val:.4f}" for val in summary.coefficientStandardErrors])  
print("T-Values:", [f"{val:.4f}" for val in summary.tValues])  
print("P-Values:",[f"{val:.4f}" for val in summary.pValues])  
  
  
print(f"\nDispersion: {summary.dispersion:.4f}")  
print(f"Null Deviance: {summary.nullDeviance:.4f}")  
print(f"Residual DF Null: {summary.residualDegreeOfFreedomNull}")  
print(f"Deviance: {summary.deviance:.4f}")  
print(f"Residual DF: {summary.residualDegreeOfFreedom}")  
print(f"AIC: {summary.aic:.4f}")

[Stage 395:> (0 + 1) / 1][Stage 396:> (0 + 1) / 1][Stage 395:> (0 + 1) / 1] [Stage 399:> (0 + 1) / 1][Stage 399:> (0 + 1) / 1][Stage 400:> (0 + 1) / 1] [Stage 404:> (0 + 1) / 1]

--- Regression Summary ---

[Stage 405:> (0 + 1) / 1][Stage 406:> (0 + 1) / 1] [Stage 410:> (0 + 1) / 1]

Coefficient Standard Errors: ['75.3627', 'inf', '3299.2815', '893.0967', '9129.4900', '9147.3984', '9183.8577', '9186.5328', '9231.8700', '1808.4642', '2323.5808', '1339.0523', '1436.1112', '7469.8953', '7472.2938', '7487.3352', '7504.5113', '7504.4248', '7508.6605', '7528.8183', '7539.7113', '7543.0325', '7542.7247', '7548.7186', '7553.7983', '7593.7045', '7590.8310', '7616.5959', '7615.0040', '7617.2648', '7622.3009', '7620.5336', '7628.1773', '7659.8343', '7682.2817', '7697.0545', '7720.2852', '7742.4696', '7751.9989', '7831.9358', '7843.7654', '7861.8520', '7859.4928', '7933.5096', '7934.3170', '7931.3088', '8117.7557', '8042.6248', '8071.2966', '8016.3472', '8085.7212', '8108.9893', '8143.6584', '8117.4254', '8093.2312', '8313.0622', '8298.6706', '8389.4048', '8357.1194', '8587.6523', '8891.1782', '9062.1224', '9015.2956', '11939.6494']  
T-Values: ['45.6622', '0.0000', '1.9675', '-3.8479', '-2.0560', '-1.8969', '-3.0329', '-3.9961', '0.0249', '3.4711', '-5.7375', '2.0973', '4.0530', '0.7105', '1.9916', '1.2459', '1.2107', '1.0276', '0.7381', '0.6210', '1.3079', '0.5746', '0.8562', '0.9372', '0.6862', '0.2719', '1.6926', '0.9315', '0.3212', '1.0408', '0.2806', '0.6492', '1.5386', '0.7346', '0.7588', '2.2254', '0.9126', '0.7903', '0.7282', '0.4389', '1.7500', '0.8086', '0.6479', '0.7742', '0.4836', '0.8136', '0.5981', '1.6172', '1.1929', '1.2825', '0.5391', '-0.2562', '1.1190', '0.6921', '0.9614', '0.3243', '0.7181', '0.3635', '0.6372', '1.2128', '0.5635', '0.4246', '0.8885', '8.7980']  
P-Values: ['0.0000', '1.0000', '0.0491', '0.0001', '0.0398', '0.0579', '0.0024', '0.0001', '0.9801', '0.0005', '0.0000', '0.0360', '0.0001', '0.4774', '0.0464', '0.2128', '0.2260', '0.3041', '0.4605', '0.5346', '0.1909', '0.5656', '0.3919', '0.3487', '0.4926', '0.7857', '0.0906', '0.3516', '0.7481', '0.2980', '0.7790', '0.5162', '0.1239', '0.4626', '0.4480', '0.0261', '0.3614', '0.4294', '0.4665', '0.6607', '0.0801', '0.4188', '0.5171', '0.4389', '0.6287', '0.4159', '0.5498', '0.1059', '0.2329', '0.1997', '0.5899', '0.7978', '0.2632', '0.4889', '0.3364', '0.7457', '0.4727', '0.7162', '0.5240', '0.2252', '0.5731', '0.6712', '0.3743', '0.0000']  
  
Dispersion: 834648362.5995

[Stage 414:> (0 + 1) / 1][Stage 415:> (0 + 1) / 1]

Null Deviance: 14812470258074.5137  
Residual DF Null: 14533  
Deviance: 12077361806815.2754  
Residual DF: 14470

[Stage 420:> (0 + 1) / 1][Stage 421:> (0 + 1) / 1][Stage 420:> (0 + 1) / 1]

AIC: 339876.5652

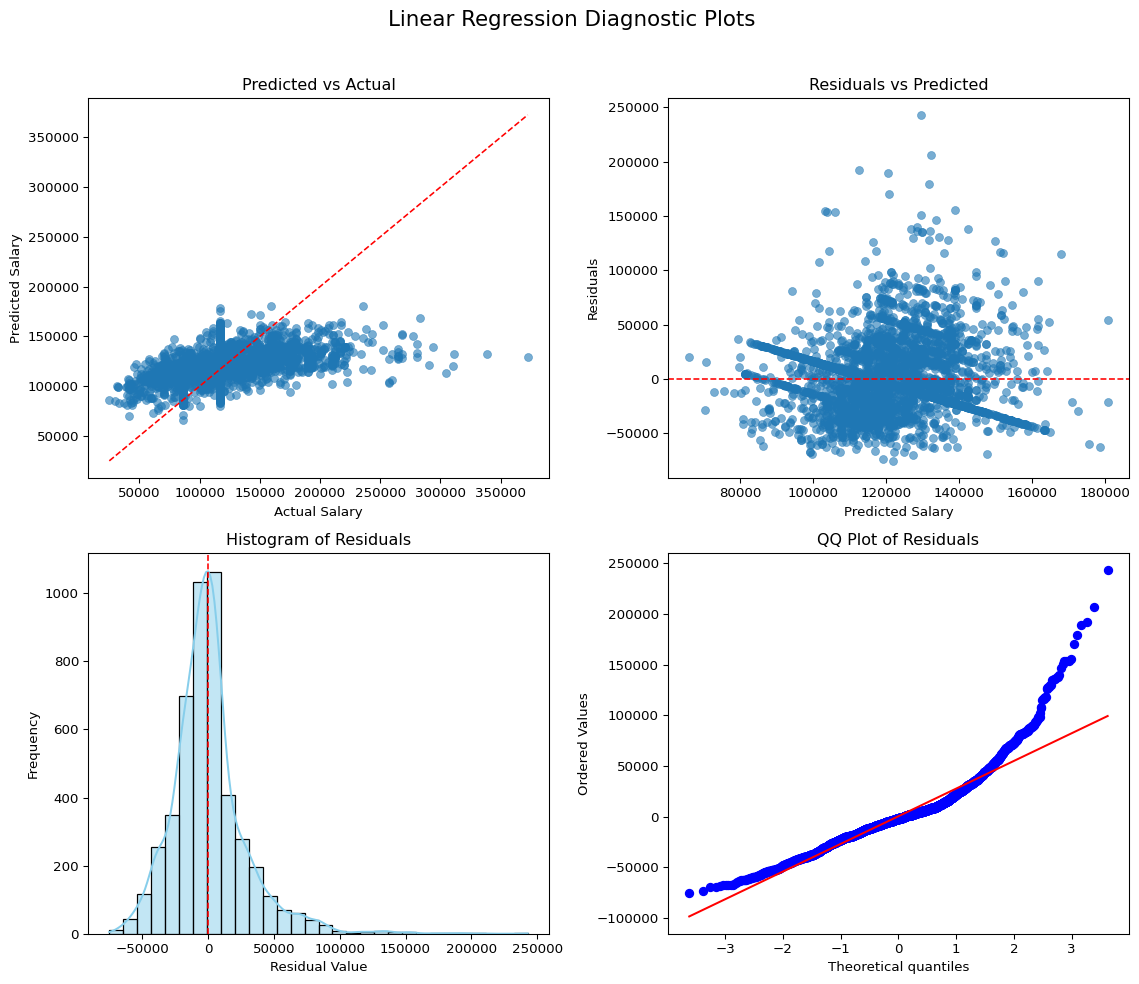
import pandas as pd  
from tabulate import tabulate  
import pandas as pd   
from IPython.display import HTML  
  
poly\_feature\_names = summary.\_call\_java("featureNames")  
poly\_coefs = [poly\_glr\_model.intercept] +list(poly\_glr\_model.coefficients)  
poly\_se = list(summary.coefficientStandardErrors)  
poly\_tvals = list(summary.tValues)  
poly\_pvals = list(summary.pValues)  
  
print("features", len(poly\_feature\_names))  
print("coefs", len(poly\_coefs))  
print("se", len(poly\_se))  
print("tvals", len(poly\_tvals))  
print("pvals", len(poly\_pvals))  
  
coef\_table = pd.DataFrame({  
 "Feature": ["Intercept"]+poly\_feature\_names,  
 "Estimate": [f"{v:.4f}" if v is not None else None for v in coefs],  
 "Std Error": [f"{v:.4f}" if v is not None else None for v in se],  
 "t-stat": [f"{v:.4f}" if v is not None else None for v in tvals],  
 "P-Value": [f"{v:.4f}" if v is not None else None for v in pvals],  
})  
  
coef\_table.to\_csv("poly\_glr\_summary.csv", index=False)  
  
  
print(coef\_table)

features 63  
coefs 64  
se 64  
tvals 64  
pvals 64  
 Feature Estimate Std Error t-stat P-Value  
0 Intercept 105044.9016 75.3627 45.6622 0.0000  
1 MIN\_YEARS\_EXPERIENCE 3441.2235 inf 0.0000 1.0000  
2 DURATION 0.0000 3299.2815 1.9675 0.0491  
3 IS\_INTERNSHIP 6491.3080 893.0967 -3.8479 0.0001  
4 COMPANY\_IS\_STAFFING -3436.5476 9129.4900 -2.0560 0.0398  
.. ... ... ... ... ...  
59 STATE\_NAME\_vec\_Alaska 5324.9531 8587.6523 1.2128 0.2252  
60 STATE\_NAME\_vec\_Vermont 10414.9407 8891.1782 0.5635 0.5731  
61 STATE\_NAME\_vec\_Montana 5010.1003 9062.1224 0.4246 0.6712  
62 STATE\_NAME\_vec\_West Virginia 3847.3576 9015.2956 0.8885 0.3743  
63 STATE\_NAME\_vec\_North Dakota 8009.6895 11939.6494 8.7980 0.0000  
  
[64 rows x 5 columns]

# Diagnostics

#Used AI, prompt and response are in the AI.txt file  
  
# ==========================================================  
# Diagnostic Plots for Linear Regression Model  
# ==========================================================  
  
import matplotlib.pyplot as plt  
import seaborn as sns  
import scipy.stats as stats  
import pandas as pd  
  
# Generate predictions  
predictions = glr\_model.transform(regression\_test)  
  
# Convert to Pandas  
pred\_pd = predictions.select("SALARY", "prediction").toPandas()  
  
# Compute residuals  
pred\_pd["residuals"] = pred\_pd["SALARY"] - pred\_pd["prediction"]  
  
# Set up 2x2 plot grid  
fig, axes = plt.subplots(2, 2, figsize=(12, 10))  
fig.suptitle("Linear Regression Diagnostic Plots", fontsize=16, y=1.02)  
  
# 1️⃣ Predicted vs Actual  
sns.scatterplot(  
 x="SALARY", y="prediction", data=pred\_pd,  
 ax=axes[0, 0], alpha=0.6, edgecolor=None  
)  
axes[0, 0].plot(  
 [pred\_pd["SALARY"].min(), pred\_pd["SALARY"].max()],  
 [pred\_pd["SALARY"].min(), pred\_pd["SALARY"].max()],  
 color="red", linestyle="--", linewidth=1.2  
)  
axes[0, 0].set\_title("Predicted vs Actual")  
axes[0, 0].set\_xlabel("Actual Salary")  
axes[0, 0].set\_ylabel("Predicted Salary")  
  
# 2️⃣ Residuals vs Predicted  
sns.scatterplot(  
 x="prediction", y="residuals", data=pred\_pd,  
 ax=axes[0, 1], alpha=0.6, edgecolor=None  
)  
axes[0, 1].axhline(0, color="red", linestyle="--", linewidth=1.2)  
axes[0, 1].set\_title("Residuals vs Predicted")  
axes[0, 1].set\_xlabel("Predicted Salary")  
axes[0, 1].set\_ylabel("Residuals")  
  
# 3️⃣ Histogram of Residuals  
sns.histplot(pred\_pd["residuals"], bins=30, kde=True, ax=axes[1, 0], color="skyblue")  
axes[1, 0].axvline(0, color="red", linestyle="--", linewidth=1.2)  
axes[1, 0].set\_title("Histogram of Residuals")  
axes[1, 0].set\_xlabel("Residual Value")  
axes[1, 0].set\_ylabel("Frequency")  
  
# 4️⃣ QQ Plot of Residuals  
stats.probplot(pred\_pd["residuals"], dist="norm", plot=axes[1, 1])  
axes[1, 1].set\_title("QQ Plot of Residuals")  
  
plt.tight\_layout()  
plt.show()

[Stage 426:> (0 + 1) / 1][Stage 426:> (0 + 1) / 1][Stage 427:> (0 + 1) / 1] [Stage 431:> (0 + 1) / 1]



# Model Evaluation

# ==========================================================  
# 6. Model Evaluation  
# ==========================================================  
  
from pyspark.ml.evaluation import RegressionEvaluator  
import matplotlib.pyplot as plt  
import seaborn as sns  
import pandas as pd  
  
# --- Evaluate on test data ---  
evaluator\_r2 = RegressionEvaluator(  
 labelCol="SALARY", predictionCol="prediction", metricName="r2"  
)  
evaluator\_rmse = RegressionEvaluator(  
 labelCol="SALARY", predictionCol="prediction", metricName="rmse"  
)  
  
r2 = evaluator\_r2.evaluate(predictions)  
rmse = evaluator\_rmse.evaluate(predictions)  
  
print("===============================================")  
print("Model Evaluation Metrics")  
print(f"R² (Coefficient of Determination): {r2:.4f}")  
print(f"RMSE (Root Mean Squared Error): {rmse:.2f}")  
print("===============================================")  
  
# --- Convert predictions to pandas for plotting ---  
pred\_pd = predictions.select("SALARY", "prediction").toPandas()  
  
# --- Predicted vs Actual Plot ---  
plt.figure(figsize=(8, 6))  
sns.scatterplot(  
 x="SALARY", y="prediction", data=pred\_pd,  
 alpha=0.6, color="steelblue", edgecolor=None, label="Predicted"  
)  
  
# Ideal fit line (y = x)  
plt.plot(  
 [pred\_pd["SALARY"].min(), pred\_pd["SALARY"].max()],  
 [pred\_pd["SALARY"].min(), pred\_pd["SALARY"].max()],  
 color="red", linestyle="--", linewidth=1.5, label="Ideal Fit (y = x)"  
)  
  
plt.title("Model Evaluation: Predicted vs Actual Salary", fontsize=14)  
plt.xlabel("Actual Salary")  
plt.ylabel("Predicted Salary")  
plt.legend()  
plt.grid(alpha=0.3)  
plt.tight\_layout()  
plt.show()

[Stage 432:> (0 + 1) / 1][Stage 433:> (0 + 1) / 1] [Stage 437:> (0 + 1) / 1] [Stage 438:> (0 + 1) / 1][Stage 438:> (0 + 1) / 1][Stage 439:> (0 + 1) / 1][Stage 438:> (0 + 1) / 1]

===============================================  
Model Evaluation Metrics  
R² (Coefficient of Determination): 0.1768  
RMSE (Root Mean Squared Error): 28481.36  
===============================================

[Stage 444:> (0 + 1) / 1][Stage 445:> (0 + 1) / 1]

