RAJALAKSHMI EDUVERSE

20-02-2025

Instructor - Gowthami Shyam

Presented by Keerthana Praveen

DATA SCIENCE PROJECT

Exploratory Data Analysis

Instructed by - Mrs.Gowthami shyam

Presented by - Keerthana Praveen

Acknowledgment

I would like to express my sincere gratitude to everyone who supported and guided me through this project. Special thanks to Mrs. Gowthami Shyam, our instructor, for her valuable teachings and to my peers for their continuous feedback and support. Their insights played a significant role in shaping my approach and understanding of data science concepts. This project has been a rewarding learning experience, helping me apply the skills learned in data science to a real-world problem. Thank you.

TABLE
of contents

Introduction

Problem Statement

The primary objective of this project is to analyze the job market dataset, clean and preprocess the data, and uncover insights regarding factors affecting salary distributions. The analysis focuses on preparing the dataset for further modeling, including feature selection and exploration of correlations with the target variable, salary.

Motivation

The job market is dynamic, and understanding the key drivers behind salary distributions can offer insights to both employers and job seekers. By analyzing features such as job title, location, and industry, we can uncover trends that help predict and evaluate salary expectations, which is valuable for both recruitment and career development.

Objectives

1 Data Cleaning and Preprocessing:

- Remove duplicates: Detect and eliminate any duplicate rows to ensure data integrity and avoid biases in model performance. Handle
- missing values: Identify and address missing values in the dataset, such as filling missing salary values with median values and dropping columns like 'Locality' that have a high percentage of missing data. Salary column cleanup: Clean the salary column by extracting the
- lower bound of the salary range, converting it into a unified format (e.g., annual salary), and ensuring consistency across the dataset. Outlier detection and removal: Detect outliers in the salary data using
- Z-scores and remove them to maintain the accuracy of the analysis and prevent skewed results.

2 :Exploratory Data Analysis (EDA):

- Data visualization: Use graphical methods such as boxplots, histograms, and bar plots to identify distributions, trends, and outliers in the dataset. This will provide insights into the data's characteristics. Correlation analysis: Investigate the relationship
- between various features and the target variable (salary per annum), using correlation matrices and visualizations to identify important predictors. Pattern identification: Identify significant patterns and trends in the data, such as which job titles or locations
- are associated with higher salaries, or which factors contribute most to salary predictions.

3 Feature Selection:

- Mutual information score: Use the mutual information method to assess the relevance of each feature with respect to the target variable (salary per annum). Features that are highly informative will be retained for model training, while irrelevant ones may be dropped.
- Dimensionality reduction: Reduce the number of features by selecting the most impactful ones based on the mutual information scores, ensuring a more efficient and accurate predictive model.

4 Model Preparation:

- Modeling readiness: Prepare the dataset for machine learning by ensuring that all categorical variables are encoded properly and numeric variables are scaled or standardized as needed.
- Data splitting: Split the data into training and testing sets to ensure that the model can be trained and validated effectively, without overfitting.

Data Description

Data Sources

The dataset used for this project is sourced from Kaggle, and it contains information about job market trends, job titles, locations, salary distributions, and more. The dataset provides rich features that can be explored for insights into salary predictions.

Data Characteristics

The job market dataset includes various features that are crucial for predicting salaries and understanding market trends:

1. Job Title: Represents the role of individuals, such as Software Engineer or Data Scientist. It is a categorical feature.
2. Location: Includes city and state information, which impacts salary, with metropolitan areas offering higher salaries. Categorical feature.
3. Experience & Qualification: Affects salary levels, with more experience leading to higher pay. Can be numeric or categorical.
4. Industry: Reflects the sector, influencing salary expectations, with tech and finance typically offering higher salaries. Categorical feature.
5. Salary: The target variable, initially provided in different formats (e.g., monthly or range-based), which needs to be standardized.
6. Company Size: Larger companies generally offer higher salaries. Categorical (ordinal) feature.
7. Experience Level: Indicates the career stage (entry, mid, senior), affecting salary. Categorical (ordinal).
8. Employment Type: Full-time roles usually offer higher salaries than part-time or contract positions. Categorical feature.
9. Missing Values: Some columns like 'Locality' have many missing values, which are addressed by dropping or imputing them.
10. Data Types: Numeric data (Salary, Experience), categorical data (Job Title, Location), and text data (Job Title, Industry) require preprocessing and encoding for model input.

Methodology

Apporach and Algorithm

The following steps were used in this project:

- Data Preprocessing: Cleaning the dataset by removing duplicates, handling missing values, and correcting salary formatting issues.

- Exploratory Data Analysis (EDA): Conducting various visual and statistical analyses to understand the relationships and trends within the dataset.

- Feature Selection: Using mutual information regression to identify the most important features that influence salary predictions.

- Model Development: This will be the next phase after the EDA, where we will apply machine learning algorithms for salary prediction.

Evaluation Metrics

For this project, the evaluation of model performance will focus on:

- Correlation: Analyzing the relationship between features and the target variable (Salary).
- Mutual Information: This will be used for feature selection to identify the most relevant features affecting salary.
- Visualizations: Correlation heatmaps and box plots to visually assess data relationships and outliers.

Exploratory Data Analysis (EDA)

Step 1: Data Loading and Inspection

```
import pandas as pd
df = pd.read_csv('job_market.csv')
df.head(), df.info()
```

Step 2: Removing Duplicates

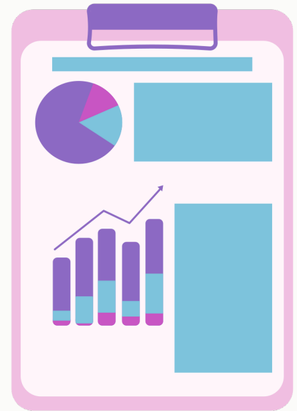
```
df = df.drop_duplicates()
```

Step 3: Handling Missing Data

- Removed the 'Locality' column due to a high percentage of missing data.
- Filled missing values in the 'Monthly Salary' column with the median value.

Step 4: Salary Data Cleaning

```
def extract_lower_bound(salary_str):
match = re.search(r'₹([\d,]+)', salary_str)
if match:
    return int(match.group(1).replace(',', ''))
  else:
    return np.nan
```



Evaluation Metrics

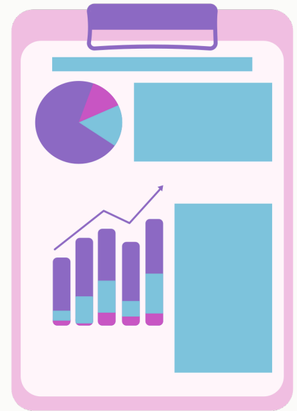
Exploratory Data Analysis (EDA)

Step 5: Handling Outliersimport pandas as pd
from scipy import stats
z_scores = np.abs(stats.zscore(df['Salary per annum']))
df_cleaned = df[z_scores < 1.5]

Step 6: Feature Encoding
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
columns_to_encode = ['Job Title', 'Location', 'City', 'State']
for col in columns_to_encode:
df_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])

Step 7: Correlation Analysis
print(df_cleaned.corr()['Salary per annum'].drop('Salary per annum').sort_values(ascending=False))

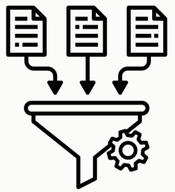
Step 8: Mutual Information for Feature Selection
from sklearn.feature_selection import mutual_info_regression
mi_scores = mutual_info_regression(X, y)



Expected Outcomes

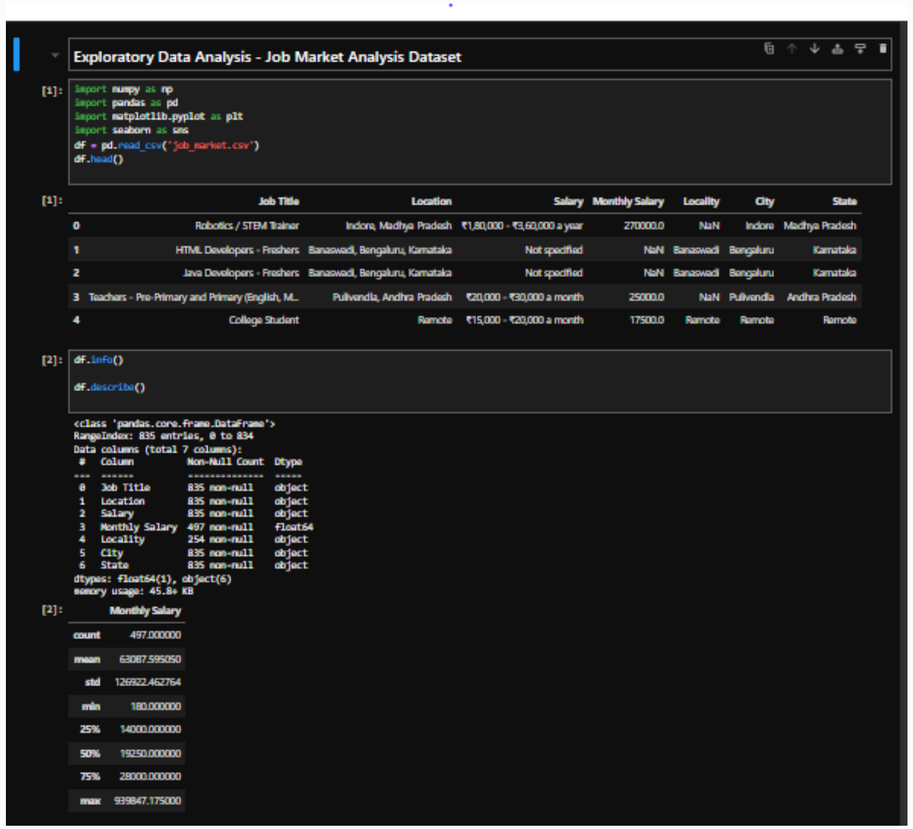
- This project will:
- Provide a clean and preprocessed dataset ready for machine learning modeling.
- Identify the most influential factors affecting salary predictions using feature selection techniques.
- Offer insights into how different job titles, locations, and other features correlate with salary expectations.

Conclusion



In this project, we focused on analyzing the job market dataset to gain insights into factors influencing salaries across various job titles, locations, and industries. Through careful data cleaning, handling of missing values, and outlier detection, we ensured that the data was ready for meaningful analysis. Key findings included the identification of strong correlations between certain features (such as Job Title and Location) with salary, as well as the successful application of feature selection techniques like mutual information to refine the dataset for model training. By addressing data inconsistencies and applying necessary transformations, we significantly improved the quality of our dataset, paving the way for accurate modeling. The results from the exploratory data analysis (EDA) and feature selection will guide the development of predictive models aimed at forecasting salary trends based on job market variables. This lays the foundation for future work in machine learning model building and evaluation, ensuring that insights derived from the data can be effectively used to guide hiring, salary benchmarking, and job market analysis. With further refinement of the models and tuning of hyperparameters, we anticipate achieving even more accurate predictions that could be used to inform job seekers, companies, and policy makers alike. Overall, the project offers valuable insights into the job market dynamics and sets the stage for more advanced predictive analytics in this domain.

Code



Exploratory Data Analysis - Job Market Analysis Dataset

```
[1]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     df = pd.read_csv('job_market.csv')
     df.head()
```

[1]:

	Job Title	Location	Salary	Monthly Salary	Locality	City	State
0	Robotics / STEM Trainer	Indore, Madhya Pradesh	₹1,80,000 - ₹3,60,000 a year	270000.0	NaN	Indore	Madhya Pradesh
1	HTML Developers - Freshers	Banaswadi, Bengaluru, Karnataka	Not specified	NaN	Banaswadi	Bengaluru	Karnataka
2	Java Developers - Freshers	Banaswadi, Bengaluru, Karnataka	Not specified	NaN	Banaswadi	Bengaluru	Karnataka
3	Teachers - Pre-Primary and Primary (English, M...	Pulivendla, Andhra Pradesh	₹20,000 - ₹30,000 a month	25000.0	NaN	Pulivendla	Andhra Pradesh
4	College Student	Remote	₹15,000 - ₹20,000 a month	17500.0	Remote	Remote	Remote

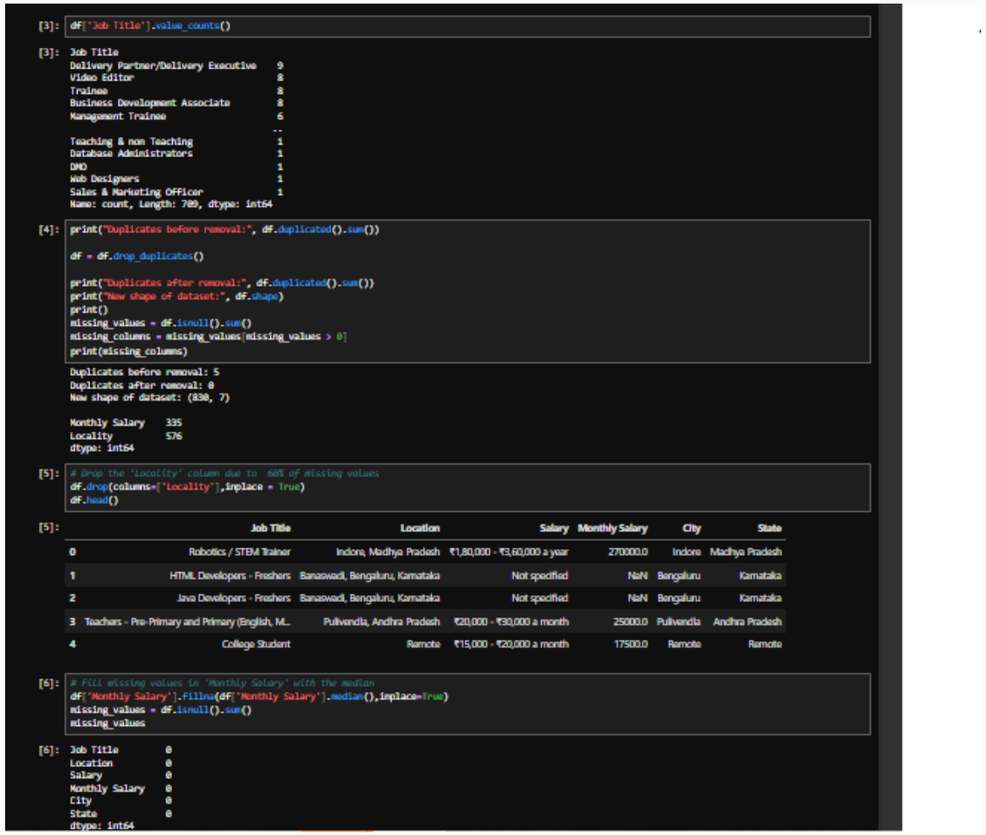
```
[2]: df.info()

     df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 835 entries, 0 to 834
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Job Title       835 non-null    object
 1   Location        835 non-null    object
 2   Salary          835 non-null    object
 3   Monthly Salary  497 non-null    float64
 4   Locality        254 non-null    object
 5   City            835 non-null    object
 6   State           835 non-null    object
dtypes: float64(1), object(6)
memory usage: 45.8+ KB
```

[2]:

	Monthly Salary
count	497.000000
mean	63087.595050
std	126922.462764
min	180.000000
25%	14000.000000
50%	19250.000000
75%	28000.000000
max	939847.175000



```
[3]: df['Job Title'].value_counts()
```

[3]:

```
Job Title
Delivery Partner/Delivery Executive    9
Video Editor                           8
Trainee                                8
Business Development Associate         8
Management Trainee                     6
                                      ..
Teaching & non Teaching                1
Database Administrators                1
DMO                                    1
Web Designers                          1
Sales & Marketing Officer              1
Name: count, Length: 709, dtype: int64
```

```
[4]: print("Duplicates before removal:", df.duplicated().sum())

     df = df.drop_duplicates()

     print("Duplicates after removal:", df.duplicated().sum())
     print("New shape of dataset:", df.shape)
     print()
     missing_values = df.isnull().sum()
     missing_columns = missing_values[missing_values > 0]
     print(missing_columns)
```

```
Duplicates before removal: 5
Duplicates after removal: 0
New shape of dataset: (830, 7)

Monthly Salary    335
Locality          576
dtype: int64
```

```
[5]: # Drop the 'Locality' column due to  60% of missing values
     df.drop(columns=['Locality'],inplace = True)
     df.head()
```

[5]:

	Job Title	Location	Salary	Monthly Salary	City	State
0	Robotics / STEM Trainer	Indore, Madhya Pradesh	₹1,80,000 - ₹3,60,000 a year	270000.0	Indore	Madhya Pradesh
1	HTML Developers - Freshers	Banaswadi, Bengaluru, Karnataka	Not specified	NaN	Bengaluru	Karnataka
2	Java Developers - Freshers	Banaswadi, Bengaluru, Karnataka	Not specified	NaN	Bengaluru	Karnataka
3	Teachers - Pre-Primary and Primary (English, M...	Pulivendla, Andhra Pradesh	₹20,000 - ₹30,000 a month	25000.0	Pulivendla	Andhra Pradesh
4	College Student	Remote	₹15,000 - ₹20,000 a month	17500.0	Remote	Remote

```
[6]: # Fill missing values in 'Monthly Salary' with the median
     df['Monthly Salary'].fillna(df['Monthly Salary'].median(),inplace=True)
     missing_values = df.isnull().sum()
     missing_values
```

[6]:

```
Job Title         0
Location          0
Salary            0
Monthly Salary    0
City              0
State             0
dtype: int64
```

Code

[7]:
```
#cleaning the salary column - removing the unnecessary stuff
import re

def extract_lower_bound(salary_str):
    match = re.search(r'₹([\d,]+)', salary_str) # \d-matches any digit (0-9). ₹ Looks for the rupee symbol.
    if match:
        return int(match.group(1).replace(',', ''))
    else:
        return np.nan

df['Salary Lower Bound'] = df['Salary'].apply(lambda x: extract_lower_bound(x) if isinstance(x, str) else np.nan)
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 830 entries, 0 to 834
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Job Title          830 non-null    object
 1   Location           830 non-null    object
 2   Salary             830 non-null    object
 3   Monthly Salary     830 non-null    float64
 4   City               830 non-null    object
 5   State              830 non-null    object
 6   Salary Lower Bound  595 non-null    float64
dtypes: float64(2), object(5)
memory usage: 51.9+ KB
None
```

[8]:
```
# Function to check if salary string contains 'month'
def convert_to_annual(salary, lower_bound):
    if isinstance(salary, str) and 'month' in salary.lower():
        return lower_bound * 12
    return lower_bound

# Apply the conversion
df['Salary Lower Bound'] = df.apply(lambda row: convert_to_annual(row['Salary'], row['Salary Lower Bound']), axis=1)

# Display the updated column
print(df[['Salary', 'Salary Lower Bound']].head())
```

```
                                Salary  Salary Lower Bound
0  ₹1,80,000 - ₹3,60,000 a year          180000.0
1                 Not specified               NaN
2                 Not specified               NaN
3     ₹20,000 - ₹30,000 a month          240000.0
4     ₹15,000 - ₹20,000 a month          180000.0
```

[9]:
```
#dropping the old salary column
df = df.drop(['Salary'],axis  = 1)
df.head()
```

[9]:

	Job Title	Location	Monthly Salary	City	State	Salary Lower Bound
0	Robotics / STEM Trainer	Indore, Madhya Pradesh	27000.0	Indore	Madhya Pradesh	180000.0
1	HTML Developers - Freshers	Banaswadi, Bengaluru, Karnataka	19250.0	Bengaluru	Karnataka	NaN
2	Java Developers - Freshers	Banaswadi, Bengaluru, Karnataka	19250.0	Bengaluru	Karnataka	NaN
3	Teachers - Pre-Primary and Primary (English, M...	Pulivendla, Andhra Pradesh	25000.0	Pulivendla	Andhra Pradesh	240000.0

[36]:
```
#renaming the new modified column
df.rename(columns={'Salary Lower Bound': 'Salary per annum'}, inplace=True)
df['Salary per annum'].fillna(df['Salary per annum'].median(),inplace=True)

df.head()
```

[36]:

	Job Title	Location	Monthly Salary	City	State	Salary per annum
0	Robotics / STEM Trainer	Indore, Madhya Pradesh	27000.0	Indore	Madhya Pradesh	180000.0
1	HTML Developers - Freshers	Banaswadi, Bengaluru, Karnataka	19250.0	Bengaluru	Karnataka	180000.0
2	Java Developers - Freshers	Banaswadi, Bengaluru, Karnataka	19250.0	Bengaluru	Karnataka	180000.0
3	Teachers - Pre-Primary and Primary (English, M...	Pulivendla, Andhra Pradesh	25000.0	Pulivendla	Andhra Pradesh	240000.0
4	College Student	Remote	17500.0	Remote	Remote	180000.0

[56]:
```
#removing outliers with z-score
from scipy import stats
import numpy as np

# Compute Z-scores
z_scores = np.abs(stats.zscore(df['Salary per annum']))

# Set threshold
threshold = 1.5

# Filter out outliers
df_cleaned = df[z_scores < threshold]

# Print results
print(f"Z-score threshold: {threshold}")
print("Number of Outliers:", (z_scores >= threshold).sum())
print("Shape before outlier removal:", df.shape)
print("Shape after outlier removal:", df_cleaned.shape)
```

```
Z-score threshold: 1.5
Number of Outliers: 17
Shape before outlier removal: (830, 6)
Shape after outlier removal: (813, 6)
```

Code

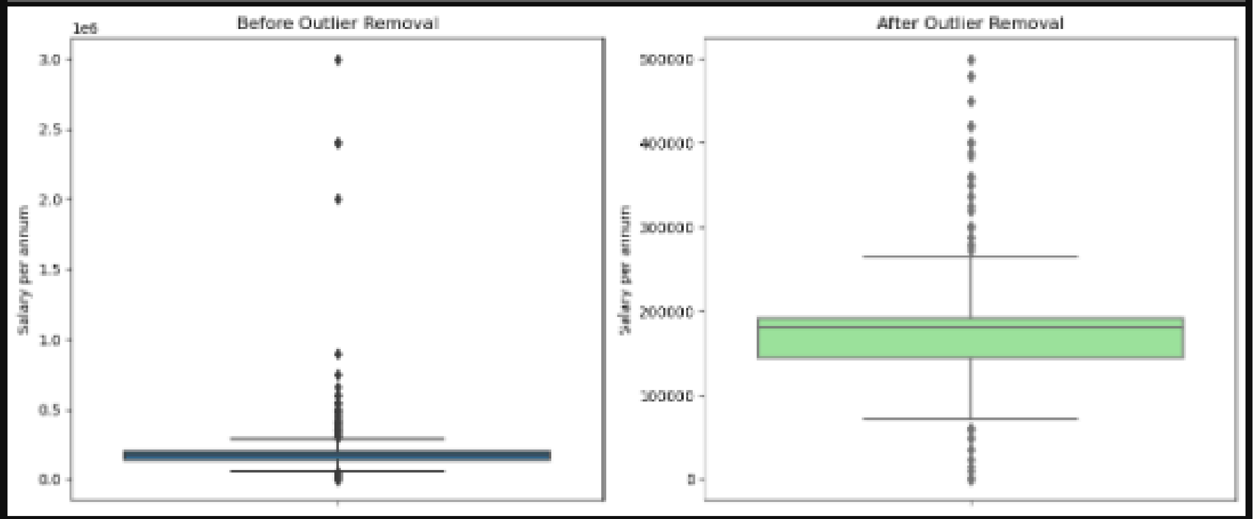
```
[57]: #visulaizing the outliers before and after
import seaborn as sns
import matplotlib.pyplot as plt

# Before and after outlier removal
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Boxplot before removing outliers
sns.boxplot(y=df['Salary per annum'], ax=axes[0])
axes[0].set_title("Before Outlier Removal")
axes[0].set_ylabel("Salary per annum")

# Boxplot after removing outliers
sns.boxplot(y=df_cleaned['Salary per annum'], ax=axes[1], color="lightgreen")
axes[1].set_title("After Outlier Removal")
axes[1].set_ylabel("Salary per annum")

plt.tight_layout()
plt.show()
```




```
[13]: #saving the cleaned dataset:
#df_cleaned.to_csv("cleaned_data.csv", index=False)
#df_cleaned.to_excel("cleaned_data.xlsx", index=False)
```

```
[59]: #encoding the categorical features with encoder
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
columns_to_encode = ['Job Title','Location', 'City', 'State']

for col in columns_to_encode:
    df_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])
```

```
C:\Users\Lenova\AppData\Local\Temp\ipykernel_14964\1113261138.py:7: 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_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])
C:\Users\Lenova\AppData\Local\Temp\ipykernel_14964\1113261138.py:7: 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_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])
C:\Users\Lenova\AppData\Local\Temp\ipykernel_14964\1113261138.py:7: 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_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])
C:\Users\Lenova\AppData\Local\Temp\ipykernel_14964\1113261138.py:7: 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_cleaned[col] = label_encoder.fit_transform(df_cleaned[col])
```

```
[48]: df_cleaned
```

[48]:

	Job Title	Location	Monthly Salary	City	State	Salary per annum
1	272	21	19250.0	13	15	180000.0
2	304	21	19250.0	13	15	180000.0
3	503	180	25000.0	120	0	240000.0
4	87	191	17500.0	129	23	180000.0
5	473	21	19250.0	13	15	180000.0
...	...	...	...	...	...	...
830	47	237	15250.0	158	17	126000.0
831	68	41	19250.0	24	4	180000.0
832	442	118	19250.0	76	28	420000.0
833	443	114	16500.0	72	16	180000.0
834	202	26	20000.0	13	15	180000.0

660 rows × 6 columns

Code

[60]:
```
#finding the correlation between target variable with all columns
print(df_cleaned.corr()['Salary per annum'].drop('Salary per annum').sort_values(ascending=False))
```

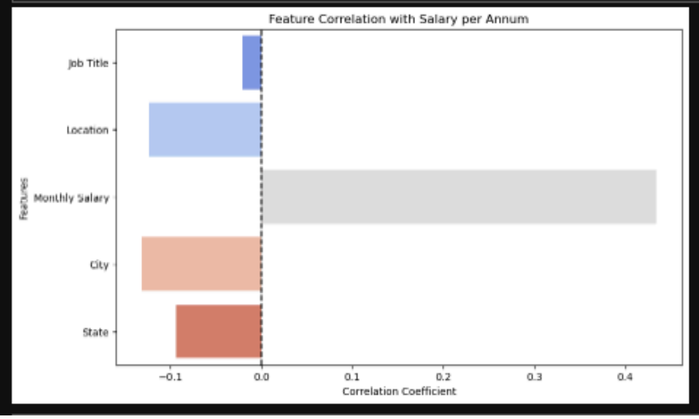
```
Monthly Salary    0.433930
Job Title         0.028974
State             0.009348
Location          0.123280
City              0.131819
Name: Salary per annum, dtype: float64
```

[61]:
```
#visualize the correlations between target variable and all columns
import seaborn as sns
import matplotlib.pyplot as plt

# Compute correlation with 'Salary per annum'
correlation = df_cleaned.corr()['Salary per annum'].drop('Salary per annum')

# Create a bar plot
plt.figure(figsize=(10, 6))
sns.barplot(x=correlation.values, y=correlation.index, palette='coolwarm')

# Labels and title
plt.xlabel("Correlation Coefficient")
plt.ylabel("Features")
plt.title("Feature Correlation with Salary per Annum")
plt.axvline(0, color='black', linestyle='--')  # Reference line at zero
plt.show()
```



[45]:
```
#feature selection using mutal info method
from sklearn.feature_selection import mutual_info_regression
import pandas as pd

# Define independent (X) and dependent (y) variables
X = df_cleaned.drop(columns=['Salary per annum'])  # Drop target column
y = df_cleaned['Salary per annum']

# Compute mutual information scores
mi_scores = mutual_info_regression(X, y)

# Convert scores into a DataFrame for better visualization
mi_df = pd.DataFrame({'Feature': X.columns, 'Mutual_Info_Score': mi_scores})

# Sort by highest importance
mi_df = mi_df.sort_values(by='Mutual_Info_Score', ascending=False)

# Display the results
print(mi_df)
```

```
          Feature  Mutual_Info_Score
2  Monthly Salary           0.571337
3            City           0.096720
1        Location           0.094894
4           State           0.065288
0       Job Title           0.037290
```

In conclusion removing the least mutal score feature and start the modeling algorthims .

References

Rajalakshmi Eduverse Data Science Course – Mrs. Gowthami Shyam, Instructor

- Course: Data Science
- Institution: Rajalakshmi Eduverse

This course provided foundational knowledge and practical skills in data science, including data preprocessing, machine learning algorithms, and model evaluation.

Kaggle (2025): "Job Market Dataset." Retrieved from https://www.kaggle.com/datasets

The dataset used for this project was sourced from Kaggle and provided relevant job market data for analysis and model development.


Thank You