**Problem Statement:**

**"The goal of this project is to build a predictive model that forecasts the salaries of employees based on various factors such as their job profile, location, company, sector, and other related features. The target variable is the employee salary, and the model will be trained using historical data about employee attributes and salaries."**

* **Target Variable**: Salary (Mean Salary)
* **Input Features**: Job, Profile, Location, Sector, Company, Revenue, Experience, Skills, City, State, and other available columns.

**Columns**

* ID -ID of the job posting
* Job - Job name
* Jobs\_Group - Jobs grouped in a few categories
* Profile - Lead/Senior/Junior or none
* Remote - Remote/Hybrid or none
* Company - Company name
* Location - the language the book is written in
* City - City
* State - State
* Frequency\_Salary - year/month/week/day
* Mean\_Salary - mean salary (USD)
* Skills - Skills, titles, certification, etc required
* Sector - Sector Company
* Sector\_Group - Sector company grouped
* Revenue - Revenue size of the company
* Employee - Number of employee size of the company
* Company\_Score - Score given by the users
* Reviews - Number of reviews of the Company\_Score
* Director - Name of the Company Director
* Director\_Score - Score of the company Director
* URL - Company website

**1. Profile**

* **Description**: Lead/Senior/Junior or none.
* **Missing Data**: 63.48%
* **Reason**: This column has a significant amount of missing data. While it could help differentiate between seniority levels, the high missing percentage reduces its reliability for prediction.
* **Decision**: **Drop** (due to high missing data and questionable value for salary prediction).

**2. Remote**

* **Description**: Remote/Hybrid or none.
* **Missing Data**: 58.11%
* **Reason**: Similar to Profile, this could influence salary depending on work flexibility. However, the high missing percentage makes it unreliable.
* **Decision**: **Drop** (high missing data and uncertain contribution to salary prediction).

**3. Revenue**

* **Description**: Revenue size of the company.
* **Missing Data**: 55.10%
* **Reason**: While the revenue of a company can impact salaries (larger companies may offer higher pay), the high missing data makes it difficult to rely on this feature.
* **Decision**: **Drop** (high missing data and limited usefulness with such gaps).

**4. Employee**

* **Description**: Number of employees in the company.
* **Missing Data**: 38.50%
* **Reason**: The size of the company (in terms of employees) can influence salaries, especially when comparing large corporations vs. smaller firms. However, with 38.5% missing data, it could lead to bias if not handled properly.
* **Decision**: **Drop** (due to missing data, though it could be valuable if imputed properly).

**5. Director**

* **Description**: Name of the company director.
* **Missing Data**: 62.51%
* **Reason**: The name of the director is unlikely to significantly impact salary prediction. The high missing percentage makes it even less useful.
* **Decision**: **Drop** (high missing data and low relevance to salary prediction).

**6. Director Score**

* **Description**: Score of the company director.
* **Missing Data**: 65.94%
* **Reason**: This metric may not have a direct correlation with salary prediction, and the high missing values reduce its predictive power.
* **Decision**: **Drop** (high missing data and low relevance).

**7. URL**

* **Description**: Company website.
* **Missing Data**: 48.22%
* **Reason**: A company's website is unlikely to directly correlate with salary prediction. It also has significant missing data.
* **Decision**: **Drop** (missing data and minimal relevance to salary).

**Project Documentation: Salary Prediction Model Using Machine Learning**

**1. Problem Statement**

The objective of this project is to predict the **Mean Salary** for different job profiles in the dataset, which includes job-related information such as job titles, company details, location, sector, and various other features. This is a regression problem where we are predicting a continuous numerical value, i.e., the **Mean Salary**.

**2. Data Collection**

The data is collected in the form of a CSV file containing various job-related attributes. These attributes include:

* Job title
* Company
* Location
* Skills
* Sector
* Job profile
* Salary details (mean salary for training, target variable for prediction)

We also received test data which does not contain the **Mean Salary** column as it is used for prediction after training the model.

**3. Data Preprocessing**

Before using the data to train machine learning models, we need to clean and preprocess it to ensure the models work effectively. Here's a breakdown of what was done during data preprocessing:

* **Missing Values Handling**:
  + For categorical columns like City, State, Company, and Location, we filled missing values using the mode (most frequent value) or based on the relationship between columns (e.g., filling State based on City).
  + For numerical columns like Company\_Score and Reviews, missing values were filled using the median or zero respectively.
* **Dropping Irrelevant Columns**:
  + We removed columns that had high missing values or did not contribute to predicting the target variable, such as Profile, Remote, Revenue, Employee, Director, Director\_Score, and URL.
* **Feature Encoding**:
  + **Target Encoding** was used for high-cardinality categorical features like Job, Company, Location, and City. This technique converts these categorical variables into numerical representations based on the target variable.
  + **One-Hot Encoding** was applied to moderate-cardinality categorical variables like Jobs\_Group, State, Frecuency\_Salary, Sector, and Sector\_Group to convert them into binary (0/1) columns.
  + **TF-IDF Encoding** was used to process the Skills column, which contains a list of skills as a string. TF-IDF is useful for representing the relevance of each word in a document relative to other words in the dataset.
* **Reindexing**:
  + The training and test data sets were reindexed to ensure that the test data has the same structure (features) as the training data. This is important for consistency when making predictions.

**4. Model Selection and Training**

We trained three machine learning models on the preprocessed data to predict the mean salary. These models are commonly used in regression tasks, and each was evaluated on training data:

* **Linear Regression**:
  + A simple regression model that assumes a linear relationship between the features and the target variable. It is quick to train and serves as a baseline model.
* **Random Forest Regressor**:
  + A powerful ensemble learning method that uses multiple decision trees. It is known for handling non-linear relationships well and is more robust against overfitting compared to single decision trees.
* **XGBoost**:
  + A popular gradient boosting algorithm known for its high performance and efficiency. It builds decision trees sequentially and optimizes model accuracy through boosting.

**5. Model Evaluation**

Each of the models was evaluated using the following metrics:

* **Mean Absolute Error (MAE)**: Measures the average magnitude of errors in the predictions, without considering their direction.
* **Mean Squared Error (MSE)**: Measures the average squared differences between predicted and actual values, giving higher weight to larger errors.
* **R-squared (R2)**: Represents the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R2 value indicates a better fit.

The **Random Forest Regressor** gave the best results based on these evaluation metrics, outperforming the other models.

**6. Model Predictions**

Once the best-performing model was identified (Random Forest Regressor), it was used to predict salaries for the test data. Since the test data did not include the target variable, the predictions were made on this unseen data.

**7. Model Serialization (Pickling)**

After training the best model, we saved it using **Pickle**, a Python library for object serialization. This allows us to save the trained model and load it at a later time without retraining. This is particularly useful for deploying models into production or sharing them with others.

**8. Conclusion**

* This project successfully demonstrated how to process job-related data and use machine learning models to predict salaries.
* We learned the importance of data preprocessing, feature engineering, and model evaluation for accurate predictions.
* The **Random Forest Regressor** was the best-performing model, offering high accuracy in predicting salaries.

**Future Improvements**

* **Hyperparameter Tuning**: Further optimization of the models by fine-tuning the hyperparameters to improve prediction accuracy.
* **Feature Engineering**: Additional features like job experience, education level, etc., could be explored to further enhance the model.
* **Model Deployment**: The trained model could be deployed as a web service, allowing real-time predictions.

**Key Takeaways**

* Data preprocessing and feature encoding are critical steps before training machine learning models.
* Model evaluation using multiple metrics helps identify the most suitable model for the task.
* Pickling allows the model to be saved and reused without retraining, saving time and resources.