

Assignment: 2

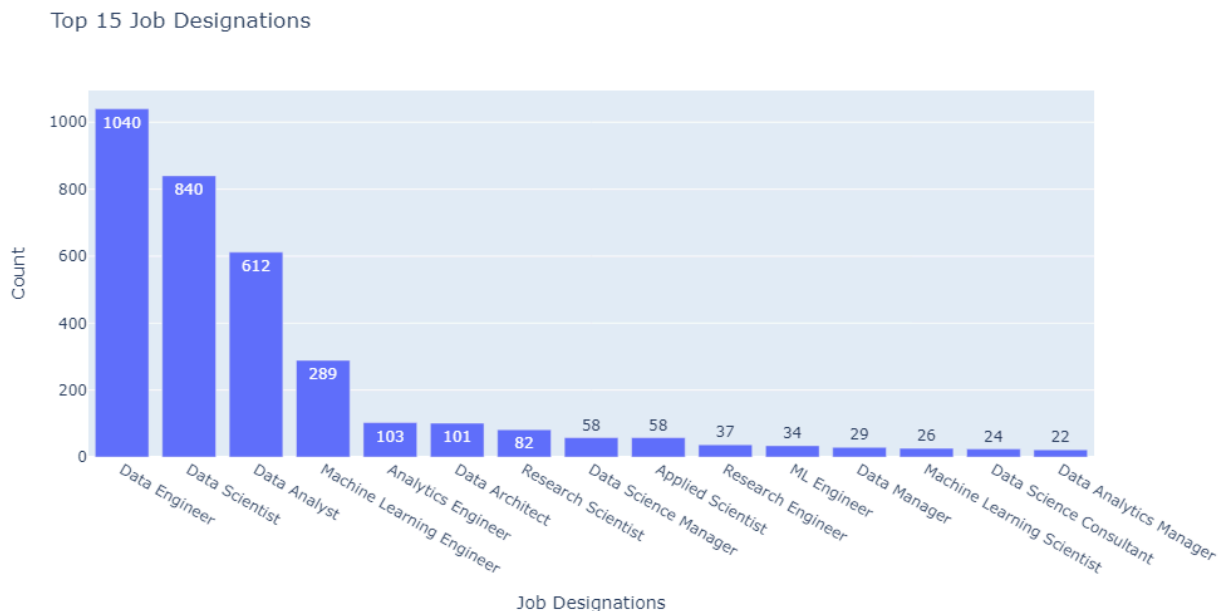
1. Data Exploration and Preprocessing:

- **Data Loading**

The initial step in our analysis involved loading the Data Science Salaries 2023 dataset. This dataset, sourced from Kaggle, provides a comprehensive overview of salary trends in the data science field for the year 2023. The data includes various features such as job titles, locations, experience levels, and employment types, all of which are crucial for understanding the factors influencing salaries.

- **Exploratory Data Analysis (EDA)**

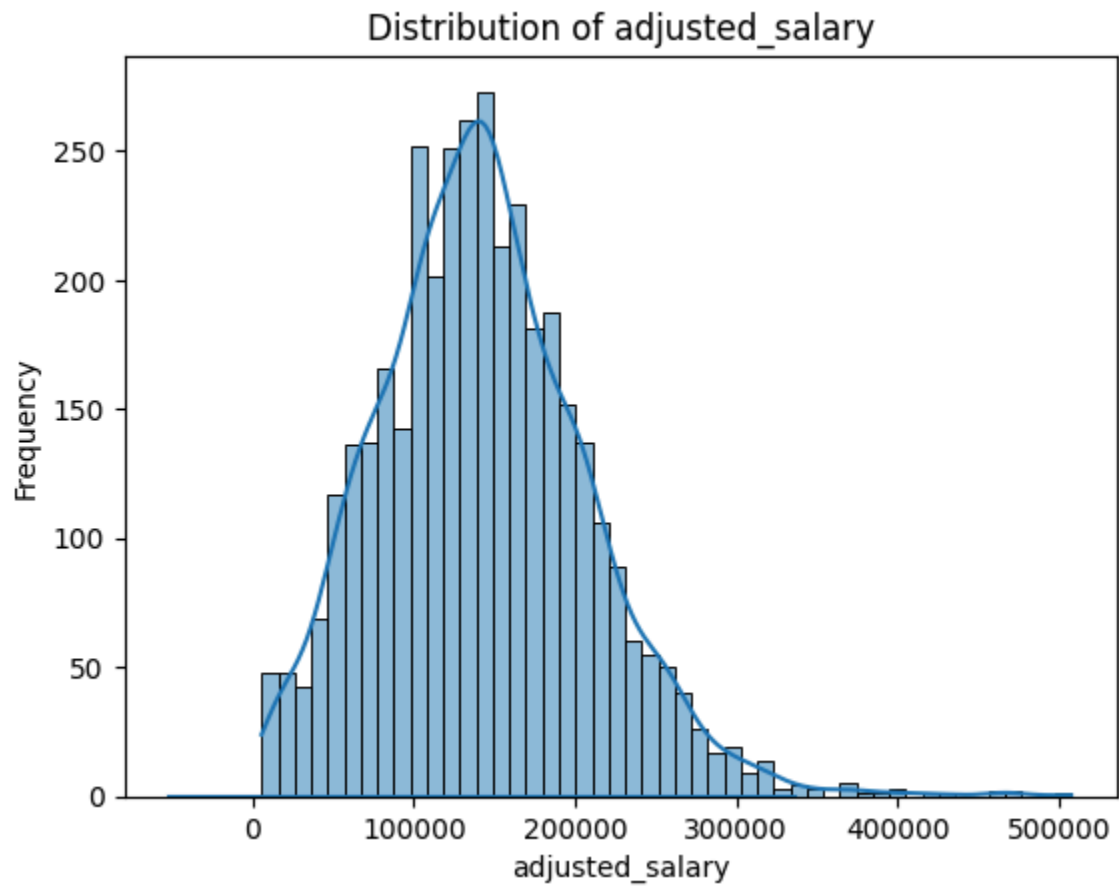
Exploratory Data Analysis (EDA) is a critical step in understanding the dataset's underlying structure and identifying any patterns, anomalies, or relationships. We began by computing summary statistics to get a high-level overview of the dataset. This included measures such as mean, median, standard deviation, and quartiles for numerical features. These statistics helped us understand the central tendency, dispersion, and overall distribution of the data.



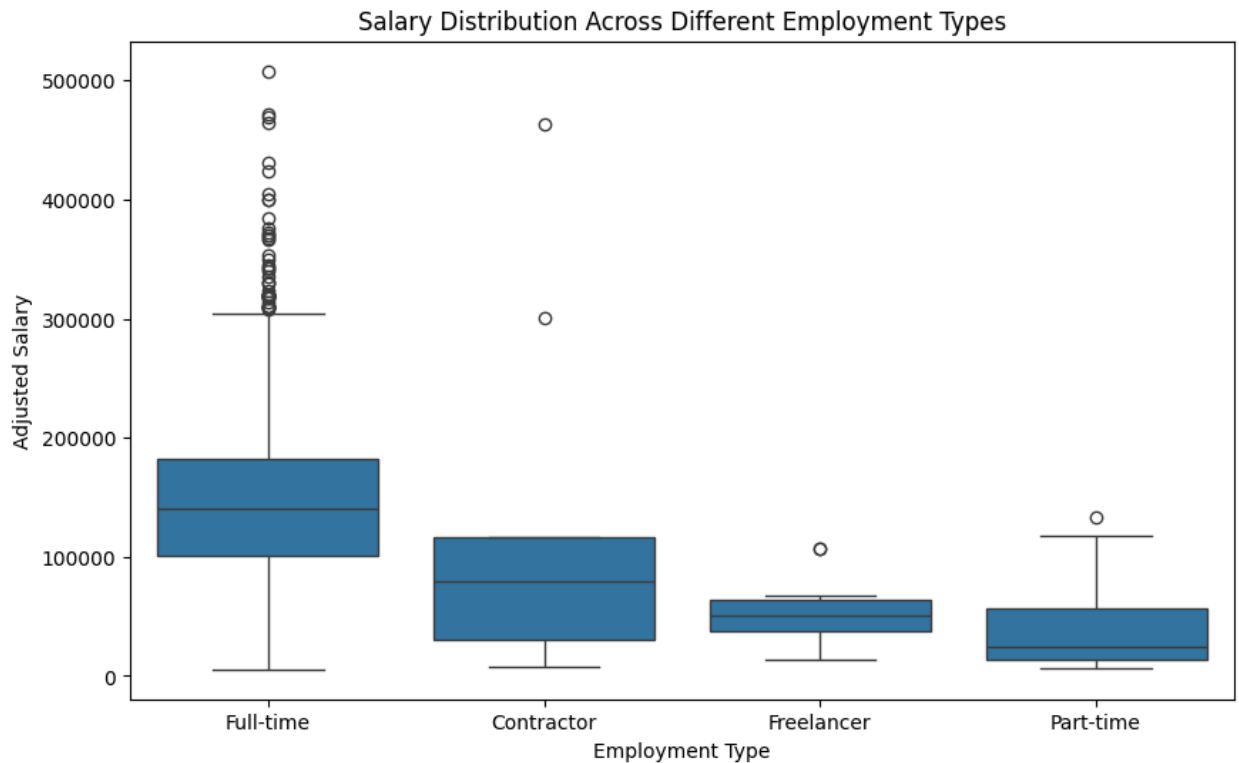
Visualizations

Visualizations are powerful tools for uncovering insights that may not be immediately apparent from raw data. Several types of visualizations were used:

- ❖ **Histograms:** These were used to visualize the distribution of adjusted salary. Histograms provided a clear view of the frequency distribution, allowing us to identify skewness and potential outliers.



- ❖ **Box Plots:** Box plots were employed to highlight the spread and skewness of salary data, as well as to identify outliers.



Data Cleaning and Handling Missing Values

Data cleaning is an essential step to ensure the quality and integrity of the dataset. Here's a summary of the cleaning process:

Handling Missing Values

Missing values can significantly impact the performance of machine learning models. In this dataset, missing values were handled using forward fill, where missing entries are replaced with the preceding non-missing value. This method was chosen as it maintains the sequence of the data.

Outliers

Outliers can skew the results and lead to misleading insights. Outliers in the numerical features were identified and handled appropriately. For instance, extremely high or low salary values were examined to determine if they were data entry errors or genuine outliers. If deemed necessary, these outliers were removed or transformed.

Encoding Categorical Variables

Machine learning algorithms typically require numerical input. Therefore, categorical variables in the dataset were converted into numerical values using Label Encoding. This process involves assigning each unique category in a feature a numerical value. For instance, the 'employment_type' feature, which includes categories like 'Full-time', 'Part-time', 'Contract', and 'Freelance', was encoded numerically.

Splitting the Data

The final preprocessing step involved splitting the dataset into training and testing sets. This is a crucial step for evaluating the performance of our machine learning models. The data was split in such a way that 80% was used for training and 20% for testing. This ensured that our models could be trained on a substantial portion of the data while still being evaluated on an unseen subset to assess their generalizability.

2. Model Training & 3. Model Selection & Optimization

Training Multiple Models

After preprocessing the data, the next step was to train multiple machine learning models to predict the salary range. The goal was to identify the best-performing model that could accurately predict the salary range based on the provided features.

Models Trained

Three different machine learning models were chosen for training:

- **Logistic Regression**
- **Random Forest Classifier**
- **Gradient Boosting Classifier**

These models were selected due to their varying complexity and ability to handle different types of data. Logistic Regression is a simple yet effective model for binary and multiclass classification problems. Random Forest and Gradient Boosting are more complex ensemble methods that often perform well on structured data.

Model Training Process

Each model was trained using the training data, and the performance was evaluated on the test data. The training process involved the following steps:

Logistic Regression

Logistic Regression is a linear model used for classification. It predicts the probability of a target variable belonging to a particular class.

- **Why Logistic Regression?**
 - Simplicity: Easy to implement and interpret.
 - Efficiency: Performs well on linearly separable data.
 - Probabilistic Approach: Provides probabilities for class membership.

Random Forest Classifier

Random Forest is an ensemble learning method that constructs multiple decision trees and merges them to get a more accurate and stable prediction.

- **Why Random Forest?**
 - Robustness: Handles missing values and outliers effectively.
 - Feature Importance: Provides insights into the importance of each feature.
 - Accuracy: Often provides high accuracy for classification tasks.

Gradient Boosting Classifier

Gradient Boosting is another ensemble technique that builds models sequentially, with each new model correcting the errors of the previous ones.

- **Why Gradient Boosting?**
 - High Performance: Often leads to better accuracy than other models.
 - Flexibility: Can be used for regression and classification tasks.
 - Control over Bias-Variance Tradeoff: Allows for fine-tuning to avoid overfitting.

MLflow for Experiment Tracking

MLflow was used to track the experiments, including parameters, metrics, and artifacts. This facilitated the comparison of different models and hyperparameters.

- **Experiment Tracking:** Each training run was logged as an experiment in MLflow. Parameters such as model type and hyperparameters, and metrics such as accuracy were recorded.

- **Model Logging:** The trained models were logged as artifacts, allowing them to be loaded and used later.

Example of an MLflow experiment tracking dashboard:

Performance Evaluation

The performance of each model was evaluated using accuracy as the primary metric. Accuracy measures the proportion of correct predictions made by the model.

Accuracy Scores

- **Logistic Regression:** Achieved an accuracy of 26.7% on the test set.
- **Random Forest Classifier:** Achieved an accuracy of 28.3% on the test set.
- **Gradient Boosting Classifier:** Achieved an accuracy of 29.5% on the test set.

These accuracy scores were compared to identify the best-performing model. The model with the highest accuracy was selected for further optimization and deployment.

Hyperparameter Tuning

The selected model was further optimized using hyperparameter tuning. Grid Search was employed to find the best combination of hyperparameters.

- **Grid Search:** A systematic approach to hyperparameter tuning that exhaustively searches through a specified parameter grid.
- **Cross-Validation:** Used in conjunction with Grid Search to evaluate each combination of parameters, ensuring the model generalizes well to unseen data.

Example hyperparameters tuned for the Random Forest model:

- **Number of Trees (n_estimators):** 100, 200
- **Maximum Depth (max_depth):** None, 10, 20

The best model after hyperparameter tuning achieved an accuracy of 29.6% on the test set, demonstrating improved performance over the initial models.

4. Streamlit Application

Building the Streamlit Application

The Streamlit application was developed to enable users to input data and predict the salary range using the trained machine learning model. The application consists of several key sections:

- **Loading the Trained Model**

The trained model, which was saved using MLflow, was loaded into the Streamlit application. This model is used to make predictions based on user inputs. The model loading process ensures that the application utilizes the best-performing model identified during the training and optimization phases.

- **Designing the User Interface**

The user interface of the Streamlit application was designed to be intuitive and user-friendly. Users can input values for various categorical features through dropdown menus. These inputs include features such as employment type, job category, experience level, employee residence, remote ratio, company location, and company size.

Each categorical feature was encoded using label encoders, which convert the categorical inputs into numerical values that the model can process. This encoding ensures that the model receives the correct format of data for making accurate predictions.

- **Making Predictions**

Once the user inputs the data, the application encodes these inputs and uses the loaded model to make predictions. The predicted salary range is then displayed to the user. This process provides an interactive and immediate way for users to see the predicted salary range based on their inputs.

On LocalHost

localhost:8502/#salary-prediction

Salary Prediction

Select employment_type

Full-time

Select job_category

Other

Select experience_level

Senior-level/Expert

Select employee_residence

ES

Select remote_ratio

Full-Remote

Select company_location

ES

Select company_size

LARGE

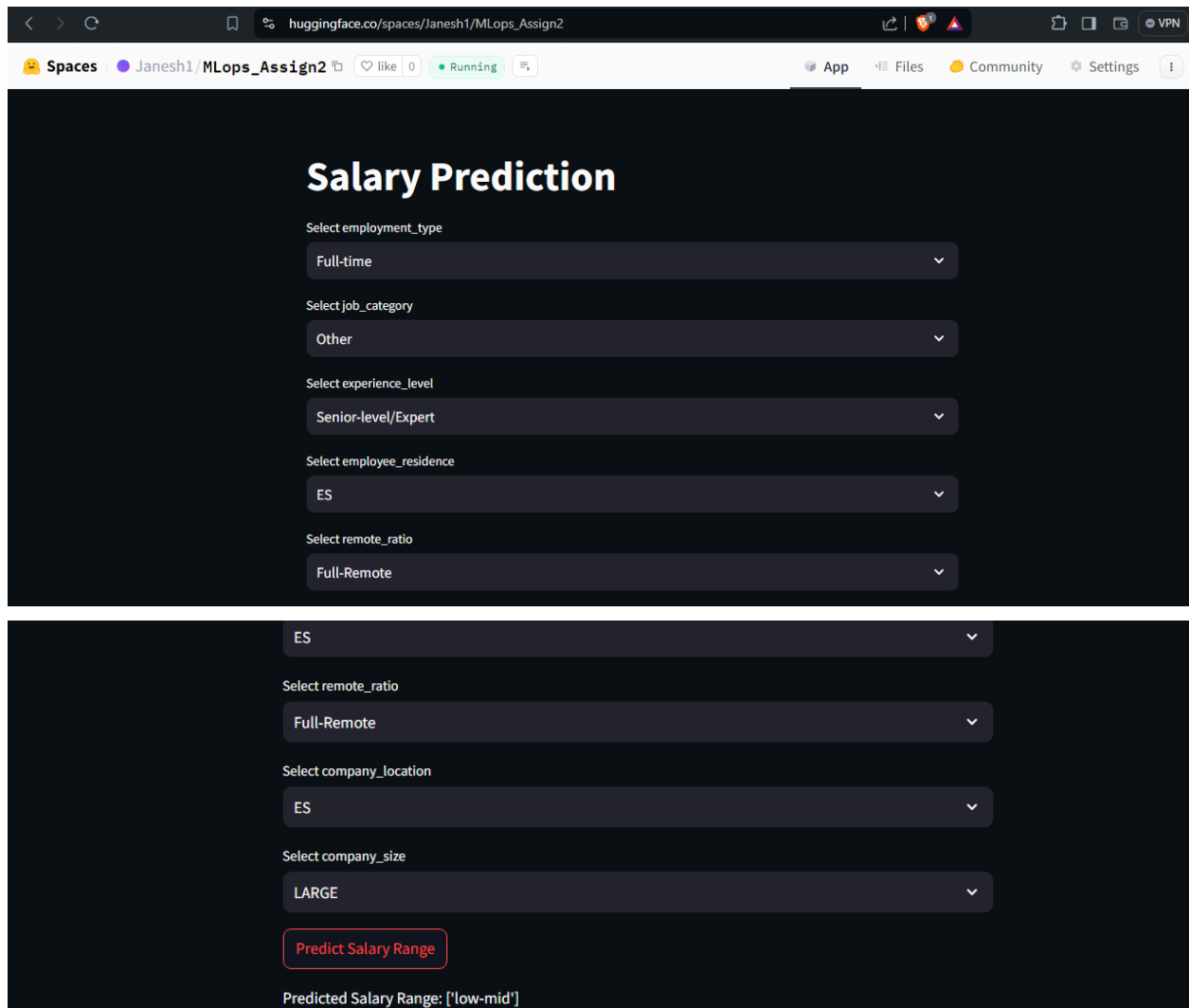
Predict Salary Range

Predicted Salary Range: ['low-mid']

5. Model Registry and deployment:

Here I have deploy the model on hugging face below is link:

https://huggingface.co/spaces/Janesh1/MLops_Assign2



The screenshot displays a web interface for a 'Salary Prediction' model deployed on Hugging Face. The interface is dark-themed and includes a header with the Hugging Face logo, the space name 'Janesh1/MLops_Assign2', and a 'Running' status indicator. The main content area features a form with seven dropdown menus for selecting input parameters: 'employment_type' (Full-time), 'job_category' (Other), 'experience_level' (Senior-level/Expert), 'employee_residence' (ES), 'remote_ratio' (Full-Remote), 'company_location' (ES), and 'company_size' (LARGE). A red-outlined button labeled 'Predict Salary Range' is positioned below the form. At the bottom, the 'Predicted Salary Range' is displayed as 'low-mid'.

Parameter	Selected Value
employment_type	Full-time
job_category	Other
experience_level	Senior-level/Expert
employee_residence	ES
remote_ratio	Full-Remote
company_location	ES
company_size	LARGE

Predicted Salary Range: ['low-mid']