Turing-Analysis

AI Job Market Insights Analysis

Project Overview

This project involves analyzing data from the AI job market, specifically focusing on factors like salary distribution, industry trends, and remote work prevalence. The dataset used includes information about job titles, salaries, industries, company sizes, AI adoption levels, and more. The goal is to extract meaningful insights about the AI job market, such as salary trends across industries, the influence of company size on salaries, and the prevalence of remote-friendly job opportunities.

Steps of Analysis

1. Data Cleaning

Before performing any analysis, we cleaned the dataset to ensure its usability. The following steps were carried out:

- **Missing Values**: We checked for missing values in the dataset. For numerical columns (e.g., `Salary_USD`), missing values were filled with the mean using NumPy (`np.nanmean()`). For categorical columns (e.g., `Industry`, `Location`), missing values were filled with the mode.
- **Data Type Conversion**: Categorical columns like `Company_Size`, `Remote_Friendly`, and `AI_Adoption_Level` were converted to the appropriate categorical data type for easier analysis and visualization.

2. Data Exploration

To gain a better understanding of the dataset, we performed some exploratory data analysis:

- **Statistical Summary**: A statistical summary of the `Salary_USD` column was generated to understand the basic distribution of salaries.
- **Visualizations**: Various plots were created to visualize the data:
  - A **histogram** showing the distribution of salaries across the dataset.
  - A **boxplot** illustrating how salaries vary by company size.
  - A **bar plot** exploring the relationship between AI adoption levels and the prevalence of remote-friendly jobs.

    - A **bar plot** showing the average salary by industry, helping us understand how different industries compensate AI professionals.

3. Feature Analysis and Engineering

Next, we delved deeper into the relationships between various features in the dataset:

- **Correlation Matrix**: We calculated a correlation matrix for numerical columns to identify any significant relationships between variables, such as salary and other numerical features.
- **Salary Bracket Creation**: We created a new feature called `Salary_Bracket` by dividing the salary data into three brackets: Low, Medium, and High. This categorization helps us analyze job distributions and trends across different salary ranges.
- **Salary Normalization**: To further enhance the analysis, we added a normalized version of the salary column using Min-Max scaling with NumPy. This allows us to compare salaries on a scale from 0 to 1.

4. Visualizing Feature Relationships

After creating new features, we visualized their relationships:

- **Job Count by Industry and Salary Bracket**: A count plot was created to show the distribution of jobs in each industry, segmented by the `Salary_Bracket`. This helps identify which industries offer higher-paying jobs more frequently.

5. Machine Learning Model Development

We developed a basic machine learning model to predict the **Salary Bracket** (Low, Medium, High) using features like `Location`, `Industry`, `Company_Size`, `AI_Adoption_Level`, and others. The following steps were followed:

- **Data Preparation**: Categorical features such as `Location`, `Industry`, and `Company_Size` were encoded into numerical values using `.cat.codes()`. This step is essential because machine learning algorithms typically require numerical input. The target variable was the newly created `Salary_Bracket` feature.
- **Train-Test Split**: We split the dataset into training and testing sets (80% for training and 20% for testing). The model was trained on the training set and evaluated on the test set.
- **Model Selection**: We used a **RandomForestClassifier**, which is an ensemble learning method suitable for classification tasks. Random forests create multiple decision trees and average their predictions for better performance.
- **Model Evaluation**: After training the model, we evaluated its performance using several metrics:
    - **Accuracy**: This measures the percentage of correct predictions made by the model.
    - **Classification Report**: This report provides precision, recall, and f1-scores for each class (Low, Medium, High). It helps understand how well the model performs for each salary bracket.
    - **Confusion Matrix**: This matrix visualizes how often instances of one class (e.g., Low salary bracket) are misclassified as another class (e.g., Medium salary bracket). The diagonal cells show correct predictions, while off-diagonal cells show misclassifications.

Model Evaluation Example

- **Accuracy**: If the model has an accuracy of 85%, it means that 85% of the predictions for the salary bracket were correct.
- **Confusion Matrix**: The matrix shows how many instances of each salary bracket (Low, Medium, High) were predicted correctly or misclassified as another bracket. The matrix helps visualize where the model might be struggling (e.g., confusing Low salaries with Medium salaries).

6. Interpretation of Results

The analysis highlighted several key findings:

- **Salary Distribution**: Salaries in the AI job market vary significantly, with certain industries like tech and finance offering higher average salaries.
- **Company Size**: Larger companies tend to offer higher salaries, as evidenced by the boxplot analysis.
- **Remote-Friendly Jobs**: The relationship between AI adoption levels and remote work showed interesting patterns, with more innovative companies appearing to embrace remote work more frequently.
- **Model Performance**: The RandomForestClassifier successfully predicted salary brackets with good accuracy, though further model tuning or feature engineering could improve performance.

7. Future Work

Moving forward, additional analysis can be conducted by:

- Expanding on feature engineering to create new meaningful variables (e.g., years of experience, education levels).
- Experimenting with other machine learning models (e.g., Logistic Regression, Support Vector Machines) to improve salary bracket prediction accuracy.
- Applying machine learning models like K-Means clustering to further analyze the relationship between salary, location, and other categorical features.

Code Documentation

Each part of the code is thoroughly commented to maintain readability and clarity. Key sections of the code contain explanations for the steps performed, such as filling missing values, creating visualizations, generating new features, and building the machine learning model. We’ve used Pandas and NumPy extensively for data handling and manipulation, Seaborn and Matplotlib for visualizations, and Scikit-Learn for machine learning.