202510-Spring 2025-ITCS-3162-001-Introduction to Data Mining-Project 3

Minthra Khounsavath

March 31, 2025

Introduction to the Problem

This project addresses the challenge of predicting employee salaries based on demographic and professional characteristics such as age, gender, education level, job title, and years of experience. Understanding these relationships can help organizations optimize salary scales and enhance fairness in pay structures. The dataset utilized for this analysis is hosted on Kaggle, and can be accessed directly through this link.

Questions to Answer:

- What factors contribute most significantly to salary variations among employees?
- How can regression analysis be utilized to predict salaries based on these factors?

Introduce the Data

The dataset for this analysis includes detailed salary information for employees at a hypothetical company. Key attributes include:

- **Age**: Numeric, representing the employee's age.
- **Gender**: Categorical, with categories 'Male' and 'Female'.

- **Education Level**: Categorical, classifications include 'Bachelor's', 'Master's', and 'PhD'.
- **Job Title**: Categorical, varies (e.g., 'Software Engineer', 'Data Analyst').
- **Years of Experience**: Numeric, total years of professional experience.
- **Salary**: Numeric, the employee's annual salary in USD.

Initial data checks revealed a dataset comprising 375 entries with minor missing values, which were subsequently cleaned resulting in 324 usable records.

Data Cleaning and Preparation

The dataset underwent several pre-processing steps to enhance its quality and ensure the accuracy of the regression models. The cleaning process involved handling duplicates, missing values, and converting categorical variables into a format suitable for modeling. I implemented the following steps in Python to prepare the data:

- **Removing Duplicate Records:** To ensure the uniqueness of each data entry, I removed any duplicate records. This step is crucial to prevent any bias that duplicates might introduce to the model.

```
    salary.drop_duplicates(inplace=True)
```

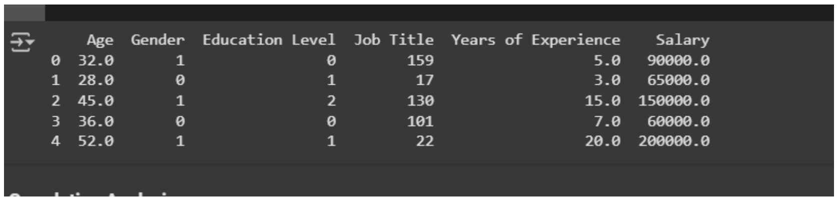
- **Handling Missing Values:** I removed rows with missing values because missing data can lead to inaccurate predictions and affect the model's performance. Given the relatively small size of the dataset, I chose deletion over imputation to maintain data integrity.

```
    salary.dropna(inplace=True)
```

- **Encoding Categorical Variables:** Since regression models require numerical input, categorical variables such as 'Gender', 'Education Level', and 'Job Title' were transformed into numeric codes. I used label encoding to convert these categories into integers.

```
from sklearn.preprocessing import LabelEncoder
label_encoders = {}
for column in ['Gender', 'Education Level', 'Job
    Title']:
    le = LabelEncoder()
    salary[column] = le.fit_transform(salary[
       column])
    label_encoders[column] = le
```

These pre-processing steps reduced the dataset from 375 to 324 entries, indicating the removal of duplicates and entries with missing data. This cleaned dataset provided a robust foundation for the subsequent regression analysis, ensuring that the models would train on accurate and representative data.



```
    Age  Gender  Education Level  Job Title  Years of Experience    Salary
0  32.0       1                0        159                  5.0   90000.0
1  28.0       0                1         17                  3.0   65000.0
2  45.0       1                2        130                 15.0  150000.0
3  36.0       0                0        101                  7.0   60000.0
4  52.0       1                1         22                 20.0  200000.0
```

Figure 1: Summary statistics of the dataset post-cleaning.

Exploratory Data Analysis

Correlation Matrix:

Correlation Analysis

To understand the relationships between the various features within our dataset and their correlation with the target variable, Salary, a correlation analysis was performed. This statistical approach helps to reveal how strongly pairs of variables are related.

Computing Correlation Matrix

Using the Pearson correlation coefficient, which measures the linear correlation between variables, the correlation matrix was computed. Values close to 1 or -1 indicate a strong positive or negative correlation, respectively, while values near 0 imply weak or no linear correlation.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate the correlation matrix
corr_matrix = salary.corr()

# Plot the correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm',
fmt=".2f", linewidths=.5)
plt.title('Correlation Matrix')
plt.show()
```

Interpretation of Correlation Coefficients

- **Age and Years of Experience:** The correlation coefficient of 0.979 indicates a very strong positive relationship, suggesting that as employees get older, they tend to have more years of experience.
- **Years of Experience and Salary:** With a correlation coefficient of 0.924, there is a strong positive relationship, indicating that salary tends to increase with the number of years one has worked.
- **Education Level and Salary:** The coefficient of 0.661900 suggests a moderate positive correlation, implying that higher educational levels might be associated with higher salaries, although other factors also play significant roles.
- **Gender and Salary:** A coefficient of 0.075420 shows a very weak correlation, indicating that gender has minimal impact on salary in this dataset, which could be indicative of a balanced or equitable pay structure.

These correlations are visualized in the correlation matrix below, which presents a comprehensive overview of how each pair of variables interrelates.

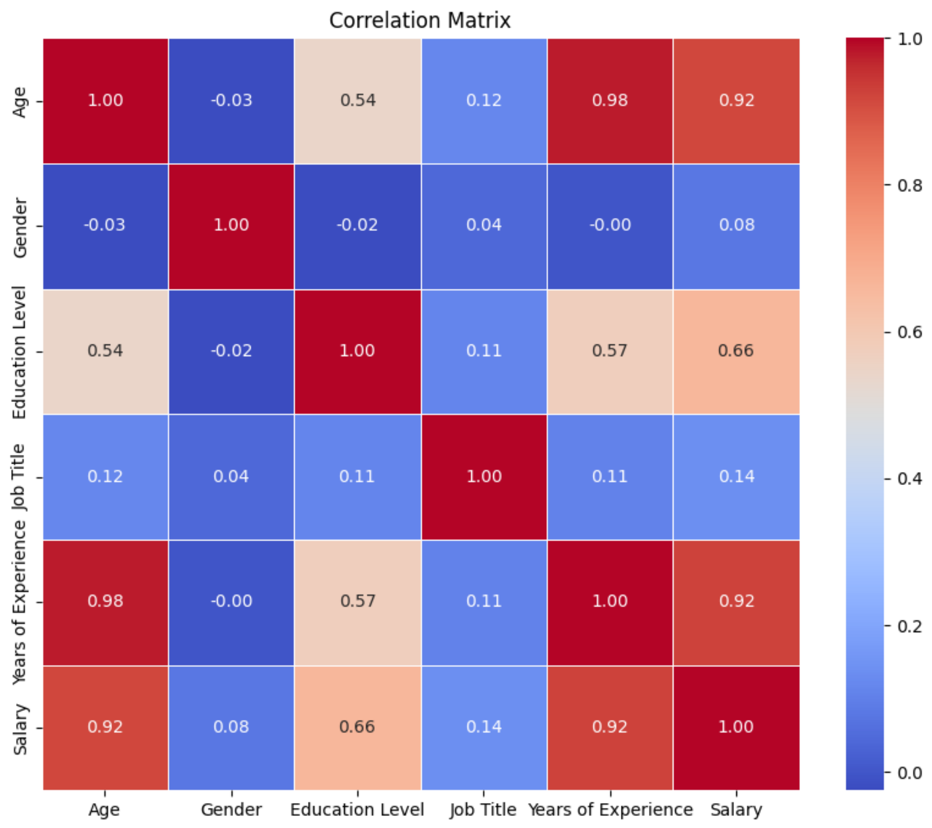


Figure 2: Correlation matrix of the variables, highlighting the relationships between different features and the salary.

Visual Exploration of Feature Relationships

To further understand the intricate dynamics between the various features in our dataset, I utilized the ‘sns.pairplot‘ function from the seaborn library. This approach provides a comprehensive visual exploration, helping to pinpoint relationships and distributions across different variables.

Generating Pair Plots

I employed the ‘sns.pairplot‘ function to generate a grid of scatter plots for each pair of variables, accompanied by histograms of each variable’s distribution on the diagonal. This method is particularly effective in illustrating both the spread and the central tendencies of the data.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Generating pair plots for a
# holistic view of the dataset
sns.pairplot(salary)
plt.show()
```

Insights from Pair Plots

From the pair plots, I derived several crucial insights that are instrumental in shaping the pre-processing and modeling strategies:

- **Linear Relationships:** The plots underscore strong linear relationships between ’Age’, ’Years of Experience’, and ’Salary’, affirming the initial findings from the correlation analysis. These relationships are characterized by clear upward trends in the scatter plots.
- **Variable Distributions:** The histograms reveal the distribution characteristics of each variable. Notably, the ’Salary’ variable is right-skewed, indicating a concentration of data points in the lower salary range with fewer instances in the higher salary brackets.
- **Outliers and Anomalies:** The visual inspection aids in identifying outliers or anomalies that may potentially skew the analysis. Observations significantly deviating from the trends could be subject to further investigation to ascertain their impact on the regression models.

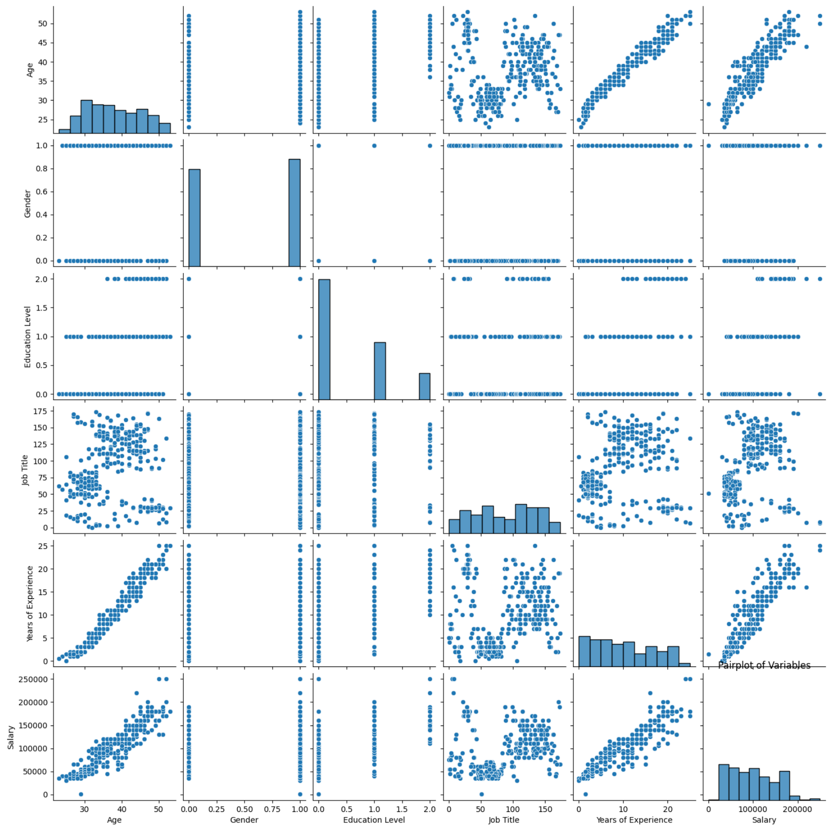


Figure 3: Comprehensive pair plots of all numerical variables, offering insights into the relationships and distributions within the dataset.

Regression Analysis and Model Evaluation

In the pursuit of understanding how well features predict employee salaries, I employed several regression models, including Linear Regression, Ridge Regression, and Lasso Regression. The aim was to find the model that best captures the complexities of our dataset while providing the most accurate predictions.

Setup and Implementation

The data was split into training and testing sets, ensuring a robust scenario for evaluating model performance. Here's a breakdown of the dataset division:

- **Training Data:** Used to fit the models, consisting of 194 records.
- **Testing Data:** Used to evaluate the models, consisting of 130 records.

Model Training

Each model was trained using the following Python code snippet, which sets up and fits the model to the training data:

```
from sklearn.linear_model
import LinearRegression, Ridge, Lasso

# Linear Regression
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Ridge Regression
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)

# Lasso Regression
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, y_train)
```

Model Performance Evaluation

Performance was primarily measured using the R-squared metric, which quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables.

Results:

- **Linear Regression**: Produced an R-squared value of 0.877, indicating that about 87.7% of the variability in salary can be explained by the model.
- **Ridge Regression**: Showed a slight improvement with an R-squared of 0.878, suggesting better handling of multi-collinearity among features.

- **Lasso Regression**: Also reported an R-squared of 0.877, similar to Linear Regression, emphasizing model simplicity and feature selection.

Visualization of Predictions

To visually assess model performance, I plotted predicted salaries against actual salaries. This scatter plot helps illustrate the accuracy of the models in predicting real-world salaries.

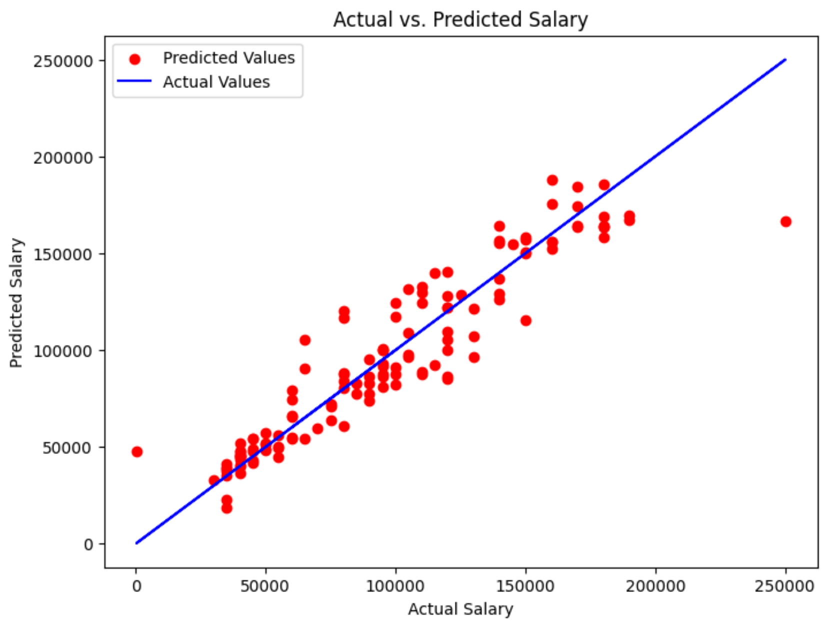


Figure 4: Scatter plot comparing actual salaries to predicted salaries from different models.

Analysis: The plot revealed that while all models performed similarly, Ridge Regression slightly outperformed the others in terms of minimizing prediction errors across higher salary ranges.

Discussion

The analysis indicated that while all models provided substantial insights into salary determinants, Ridge Regression emerged as the slightly superior model due to its ability to manage multi-collinearity. This is critical in our dataset, where features such as 'Years of Experience' and 'Age' are highly correlated.

These findings underscore the importance of choosing the right model based on the data characteristics and the specific nuances of the regression analysis. The next steps would involve refining these models further or potentially exploring more complex algorithms to enhance predictive accuracy.

Model Evaluation through Residual Analysis

In my continuous effort to validate and refine the regression models used to predict salaries, I turned to residual analysis—a critical component for understanding the effectiveness and accuracy of predictive models. This analysis helps identify non-linearity, unequal error variances, and outliers.

Residuals Explained

Residuals, the differences between observed and predicted values, provide insightful diagnostics about the regression equation. By analyzing these residuals, I could gauge whether the variance of the residuals is constant across all levels of the independent variables, a key assumption in linear regression models.

Visualizing Residuals

I utilized a residual plot, which is a graph that shows the residuals on the vertical axis and the independent variable on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, it suggests potential problems with the model used.

Interpretation of Residual Plots

The residual plot for our regression analysis revealed that the residuals do not form any discernible patterns or curves, which generally indicates that the model assumptions are met. However, there were some outliers visible, which suggests that certain predictions were significantly different from the actual values. Here are some notable discrepancies:

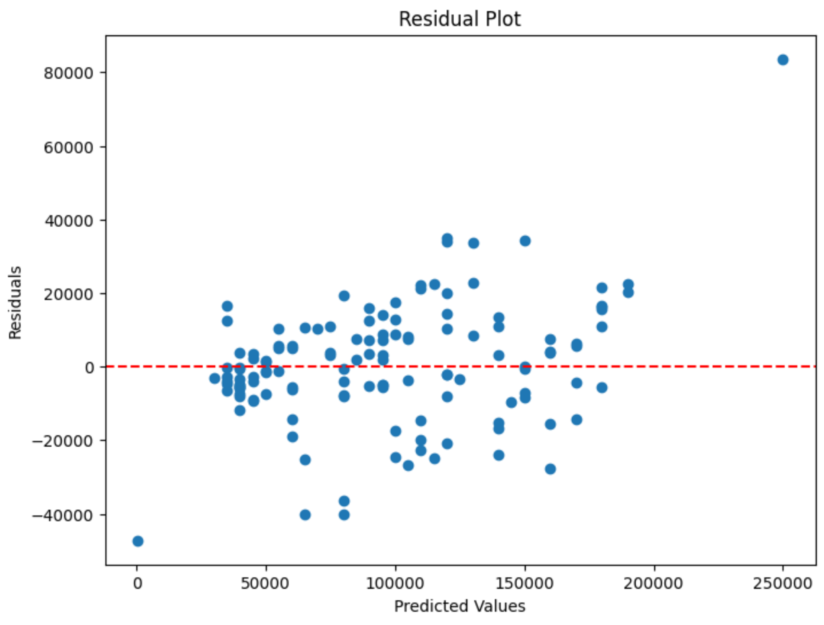


Figure 5: Residual plot for the Linear Regression model, showing deviations of predicted values from actual salaries.

- **Employee ID 132**: Predicted Salary was underestimated by $17,345.42.
- **Employee ID 108**: Predicted Salary was underestimated by $24,674.78.
- **Employee ID 9**: Predicted Salary was overestimated by $22,836.25.

Normality Test of Residuals using Q-Q Plot

Another vital step in my regression analysis was to verify the normality of the residuals. The normality assumption is a key aspect of linear regression that impacts the validity of hypothesis testing.

To assess this, I utilized a Quantile-Quantile (Q-Q) plot, which is a graphical technique for determining if two data sets come from populations with a common distribution. A Q-Q plot is a scatterplot created by plotting two sets of quantiles against one another. If both sets of quantiles come from the same distribution, we should see the points forming a line that's roughly straight.

Implementation in Python:

```
import scipy.stats as stats
# Generate a Q-Q plot
plt.figure(figsize=(8, 6))
stats.probplot(residuals, dist="norm", plot=plt)
plt.title('Q-Q Plot of Residuals')
plt.show()
```

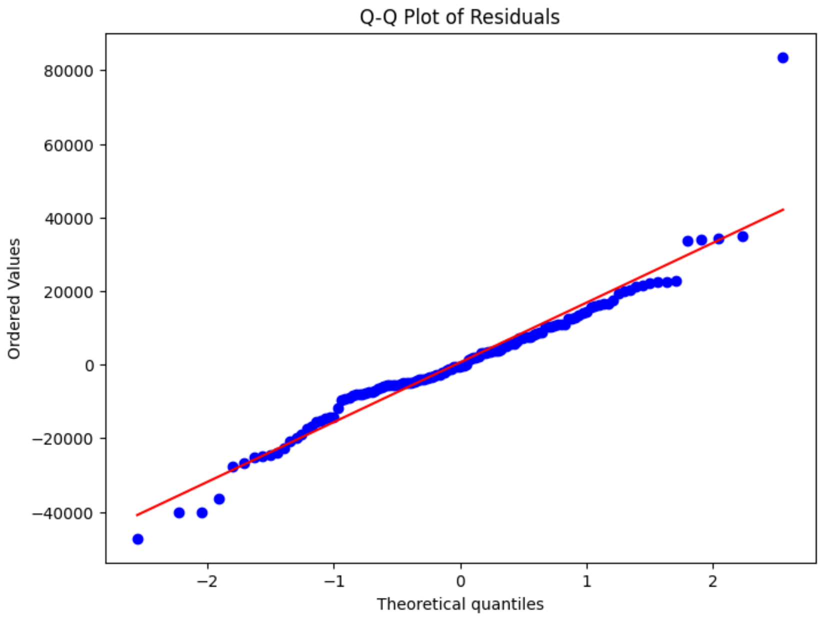


Figure 6: Q-Q Plot of Residuals indicating how closely the residuals follow a normal distribution.

Analysis of the Q-Q Plot: The resulting Q-Q plot provided a visual comparison between the observed quantiles of residuals and the expected quantiles from a standard normal distribution. The closer the points lie on the diagonal line, the more evidence there is that the data is normally distributed.

From the Q-Q plot observed, the points largely follow the line, suggesting that the residuals are approximately normally distributed. However, some deviations from the line at the tails might indicate the presence of outliers or skewness in the residual distribution.

Comparison of Regression Model Performance

To provide a clear comparison of the performance across different regression models, I used the R-squared metric, which quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables.

Python Code for Generating the Comparison Chart:

```
import matplotlib.pyplot as plt

models = ['Linear Regression', 'Ridge Regression', '
   Lasso Regression']
r2_scores = [r2, ridge_r2, lasso_r2]
print(r2_scores)
# Creating a bar chart
plt.figure(figsize=(10, 6))
plt.bar(models, r2_scores, color=['blue', 'green', 'red'
   ])
plt.xlabel('Regression Models')
plt.ylabel('R-squared Score')
plt.title('Comparison of Regression Models')
plt.show()
```

Discussion of Results: In the comparison, each model showcased its capabilities in fitting to the data:

- **Linear Regression** produced an R-squared score of 0.877, indicating a good but not perfect fit, suggesting that while the model captures a substantial proportion of the variability, there's room for improvement.
- **Ridge Regression** showed a slightly better R-squared score of 0.878, demonstrating that the addition of regularization slightly enhances the model's ability to generalize.
- **Lasso Regression** also presented an R-squared score of 0.877, similar to Linear Regression, but with the advantage of feature selection due to its regularization properties.

These findings assist in understanding how each regression technique handles the inherent variability in the dataset and its impact on the model's predictive accuracy. This

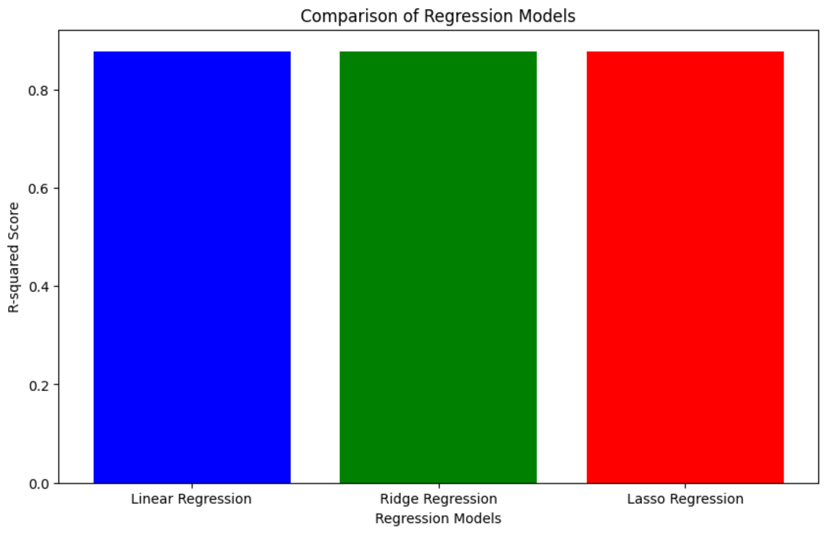


Figure 7: Bar chart comparing the R-squared scores of Linear Regression, Ridge Regression, and Lasso Regression.

Predictive Analysis and Model Application

In this section, I utilized multiple regression models to predict the salaries based on various employee characteristics. The models include Linear Regression, Ridge Regression, and Lasso Regression. I implemented a function to collect predictions from each model for a given set of employee attributes.

Python Function to Collect Predictions:

```
def collect_predictions(age, gender, education_level,
   job_title, years_of_experience):
    input_data = pd.DataFrame({
        'Age': [age],
        'Gender': [gender],
        'Education Level': [education_level],
        'Job Title': [job_title],
        'Years of Experience': [years_of_experience]
    })
    input_data['Gender'] = label_encoders['Gender'].
       transform(input_data['Gender'])
    input_data['Education Level'] = label_encoders['
```

```
       Education Level'].transform(input_data['Education
        Level'])
    input_data['Job Title'] = label_encoders['Job Title'
       ].transform(input_data['Job Title'])
    linear_pred = model.predict(input_data)[0]
    ridge_pred = ridge_model.predict(input_data)[0]
    lasso_pred = lasso_model.predict(input_data)[0]

    return (linear_pred, ridge_pred, lasso_pred)
```

Application of the Prediction Function: I applied the prediction function to four hypothetical employee profiles to illustrate how the models perform with diverse inputs. The profiles include a young professional, a mid-career professional, a senior professional, and an entry-level professional.

Visualization of Predictions: To visually compare the predictions across models, I generated a bar chart representing the predicted salaries for each profile as predicted by the three regression models.

```

# Plotting the Predictions
labels = ['Young Professional', 'Mid-career Professional
   ', 'Senior Professional', 'Entry-level Professional']
linear_preds = [pred[0] for pred in predictions]
ridge_preds = [pred[1] for pred in predictions]
lasso_preds = [pred[2] for pred in predictions]

x = np.arange(len(labels))
width = 0.25

fig, ax = plt.subplots(figsize=(12, 8))
rects1 = ax.bar(x - width, linear_preds, width, label='
   Linear Regression')
rects2 = ax.bar(x, ridge_preds, width, label='Ridge
   Regression')
rects3 = ax.bar(x + width, lasso_preds, width, label='
   Lasso Regression')

ax.set_xlabel('Profiles')
ax.set_ylabel('Predicted Salaries')
ax.set_title('Comparison of Salary Predictions by
    Regression Model')
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()

def autolabel(rects):
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}'.format(int(height)),
                    xy=(rect.get_x() + rect.get_width()
                       / 2, height),
                    xytext=(0, 3),
                    textcoords="offset points",
                    ha='center', va='bottom')

autolabel(rects1)
autolabel(rects2)
autolabel(rects3)

fig.tight_layout()

plt.show()
```

Discussion of Individual Predictions:

- For the 25-year-old female Data Analyst with a Master's degree and 2 years of experience, the predictions were closely aligned across all models, with Linear Regression predicting \$45,170.39, Ridge Regression predicting \$45,038.41, and Lasso Regression predicting \$45,170.07.

- The 35-year-old male Software Engineer with a Bachelor's degree and 10 years of experience received predictions of \$88,176.04 from Linear Regression, \$88,309.15 from Ridge Regression, and \$88,175.83 from Lasso Regression.

- For the senior profile, a 45-year-old male Senior Manager with a PhD and 20 years of experience, the predictions showed a slightly wider range: \$177,486.46 from Linear Regression, \$177,240.65 from Ridge Regression, and \$177,485.88 from Lasso Regression.

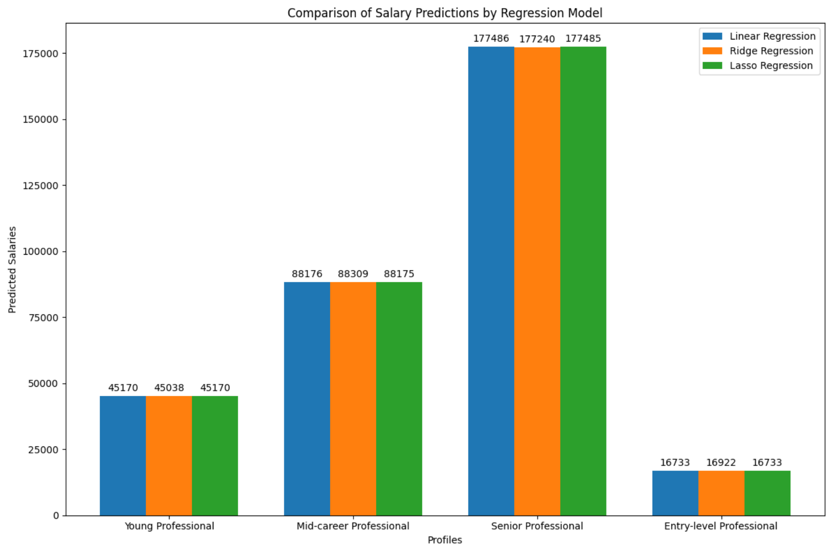


Figure 8: Bar chart comparing the predicted salaries by Linear Regression, Ridge Regression, and Lasso Regression for different professional profiles.

- The entry-level profile, a 22-year-old female Junior Marketing Coordinator with a Bachelor's degree and 1 year of experience, received predictions of \$16,733.91 from both Linear and Lasso Regressions and a slightly higher prediction from Ridge Regression at \$16,922.96.

This demonstrates the practical application of regression models in predicting salaries based on multiple factors and highlights the effectiveness of regularization in managing model complexity and prediction variance.

Conclusions

This project successfully demonstrated the utility of regression models—Linear Regression, Ridge Regression, and Lasso Regression—in predicting employee salaries based on various demographic and professional characteristics. The analysis provided valuable insights into the significant factors affecting salary variations and how these factors can be quantitatively assessed to predict salary levels.

The regression models performed robustly across the dataset, as evidenced by the R-squared values which suggest a good fit. The slight differences in performance between the models underscored the nuances of each approach:

- Linear Regression provided a baseline model with substantial predictive power.
- Ridge Regression offered slight improvements in predictive accuracy, likely due to its ability to handle multicollinearity better.
- Lasso Regression, while similar in performance to Linear Regression, added the benefit of feature selection, simplifying the model by automatically reducing the number of variables.

These models not only helped in understanding the relationship between employees' characteristics and their compensation but also highlighted the potential for human resources management to leverage statistical models for more equitable salary distributions. By integrating such models, organizations can ensure that salaries are both competitive and justifiable, based on objective criteria rather than subjective judgments.

Moreover, the exploration and visualization of the data, including correlation analysis and pair plots, provided deeper insights into the dynamics within the workforce. Such analyses help in preemptively identifying and addressing disparities in compensation that might arise due to overlooked or subtle factors.

In conclusion, the regression analysis conducted in this project offers a roadmap for organizations to enhance their compensation strategies. It encourages data-driven decision-making in salary structuring, which can lead to more motivated and satisfied employees, thereby fostering a more productive and equitable workplace environment.