https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

Introduction

The dataset Housing in London (kaggle.com) "Housing in London Yearly Variables" provides various socio-economic indicators for different boroughs in London as of December 1, 1999. It contains 1071raws and 12 variables. Each row represents a borough and includes the following key variables:

Code: Unique identifier for each borough.

Area: Name of the borough.

Date: The date of data collection (all entries are from December 1, 1999).

Median Salary: The median salary in each borough.

Mean Salary: The average salary in each borough.

Recycling Percentage: The percentage of waste recycled in each borough.

Population Size: The total population of each borough.

Number of Jobs: The total number of jobs available in each borough.

Area Size: The geographical size of the borough.

Number of Houses: The total number of houses in each borough.

Borough Flag: A binary indicator showing whether the area is a borough.

	code	area	date	median_salary	life_satisfaction	mean_salary	recycling_pct	population_size	number_of_jobs	area_size	no_of_houses	borough_flag
0	E09000001	city of london	1999-12-01	33020.0	NaN	48922	0	6581.0	NaN	NaN	NaN	1
1	E09000002	barking and dagenham	1999-12-01	21480.0	NaN	23620	3	162444.0	NaN	NaN	NaN	1
2	E09000003	barnet	1999-12-01	19568.0	NaN	23128	8	313469.0	NaN	NaN	NaN	1
3	E09000004	bexley	1999-12-01	18621.0	NaN	21386	18	217458.0	NaN	NaN	NaN	1
4	E09000005	brent	1999-12-01	18532.0	NaN	20911	6	260317.0	NaN	NaN	NaN	1

Figure 1: First 6 Raws (London Housing Yearly)

Data Cleaning and EDA

I use describe function to provide summary statistics for each column in the DataFrame

And info to display the structure of the DataFrame including data types of each column.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

```
NULL values in the DataFrame:
1955
NA values in the DataFrame:
1955
NA values in each column:
code                   0
area                   0
date                   0
median_salary         22
life_satisfaction    719
mean_salary            0
recycling_pct        211
population_size       53
number_of_jobs       140
area_size            405
no_of_houses         405
borough_flag           0
```

Figure 2: missing values in each column

I encoded categorical variables and numerical variables and set strange characters to NA.

Split the data into training and testing sets.

```
house['area'] = house['area'].astype('category', errors='ignore')
house['mean_salary'] = pd.to_numeric(house['mean_salary'], errors='coerce')
house['recycling_pct'] = pd.to_numeric(house['recycling_pct'], errors='coerce')
house['population_size'] = pd.to_numeric(house['population_size'], errors='coerce')
house['number_of_jobs'] = pd.to_numeric(house['number_of_jobs'], errors='coerce')
house['area_size'] = pd.to_numeric(house['area_size'], errors='coerce')
house['no_of_houses'] = pd.to_numeric(house['no_of_houses'], errors='coerce')
house['borough_flag'] = house['borough_flag'].astype('category', errors='ignore')
```

Figure 3

I impute all missing values using sklearn iterative imputer then compare between original dataset and the imputed one .

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

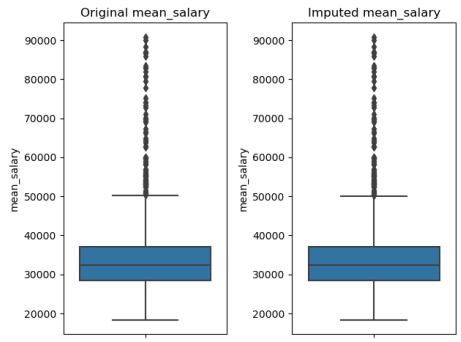


Figure 4: mean salary original vs imputed

Handling missing values.

I calculate z-scores for each numeric column to identify outliers. A threshold of 3 was set, meaning any data points with z-scores greater than 3 or less than -3 were considered outliers. These outliers were then imputed with the median value of their respective columns to ensure the dataset remained robust and not influenced by extreme values. Following this, I plot boxplots for all columns.

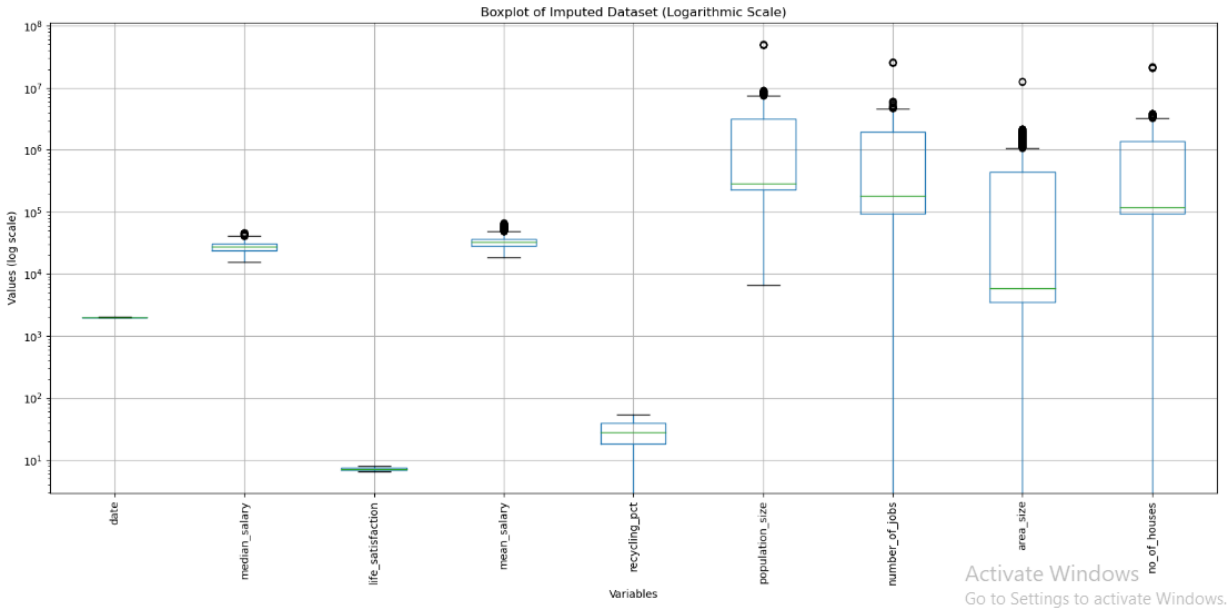


Figure 5: Boxplot of Imputed Dataset

To analyze the relationships between different variables in the dataset, we computed the correlation matrix for the numeric columns using Pearson's method. The correlation matrix provides insights into the linear relationships between pairs of variables, with values ranging from -1 to 1. Positive values indicate a direct relationship, while negative values indicate an inverse relationship.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

To visualize these correlations, we plotted a heatmap.

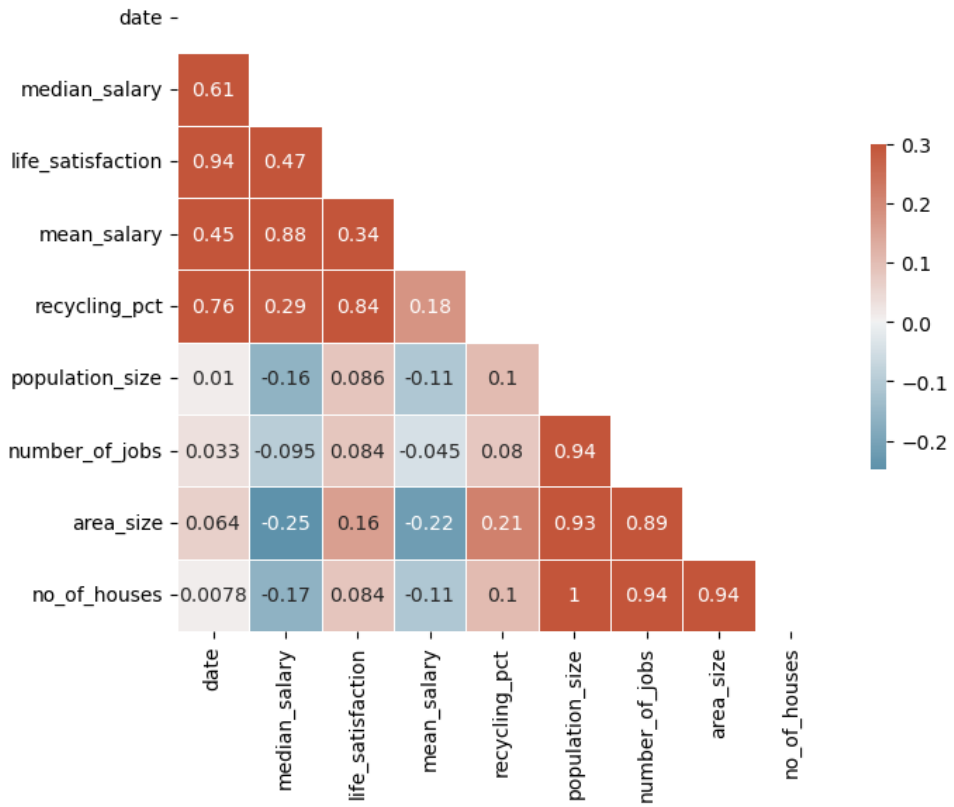


Figure 6: correlation matrix with a heatmap

We extracted the correlation values specifically for the 'mean_salary' variable to understand how it correlates with other variables in the dataset. Here is the series of correlation values for 'mean_salary' with other variables.

```
date                0.449195
median_salary       0.883123
life_satisfaction   0.338744
mean_salary         1.000000
recycling_pct       0.179909
population_size    -0.109663
number_of_jobs     -0.045110
area_size          -0.218639
no_of_houses       -0.112627
Name: mean_salary, dtype: float64
```

Figure 6: Correlation valuse for mrean salary

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

I used Variance Inflation Factor (VIF) to detect multicollinearity between the independent variables. High VIF values indicate a high correlation with other variables, which can inflate the variance of the estimated regression coefficients and make the model unstable.

$$VIF_i = \frac{1}{1 - R_i^2}$$

VIF = 1: No correlation between the independent variable and the other variables.

1 < VIF < 5: Moderate correlation, generally acceptable.

VIF > 5: High correlation, indicates multicollinearity which might need to be addressed.

Variance Inflation Factors (VIF) for the features are high, so I applied PCA to reduce dimensions and preserve as much information as possible before conducting multilinear regression.

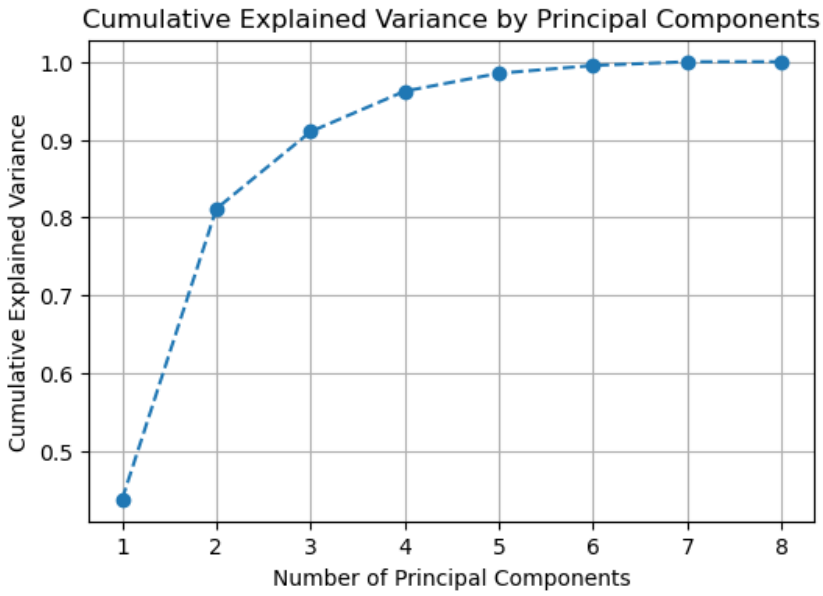
Model Selection:

1. Multilinear regression is a statistical technique used for modeling the relationship between multiple independent variables and a single dependent variable. It assumes a linear relationship between the independent variables and the dependent variable.

Multilinear regression navigates through the multidimensional space of predictors to determine the best-fitting hyperplane that minimizes the difference between observed and predicted values of the dependent variable. This makes it particularly useful for understanding how changes in one or more predictors affect the outcome, as each coefficient represents the change in the dependent variable associated with a one-unit change in the corresponding predictor, holding all other predictors constant.
$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \varepsilon$.

I applied PCA and take first 4 components, they explain more than 90% of the variance.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git



	Principal Component	Explained Variance Ratio
0	PC1	0.437555
1	PC2	0.373518
2	PC3	0.099055
3	PC4	0.052094
4	PC5	0.022975
5	PC6	0.009997
6	PC7	0.004792
7	PC8	0.000014

Figure 7: Explained variance by principal components

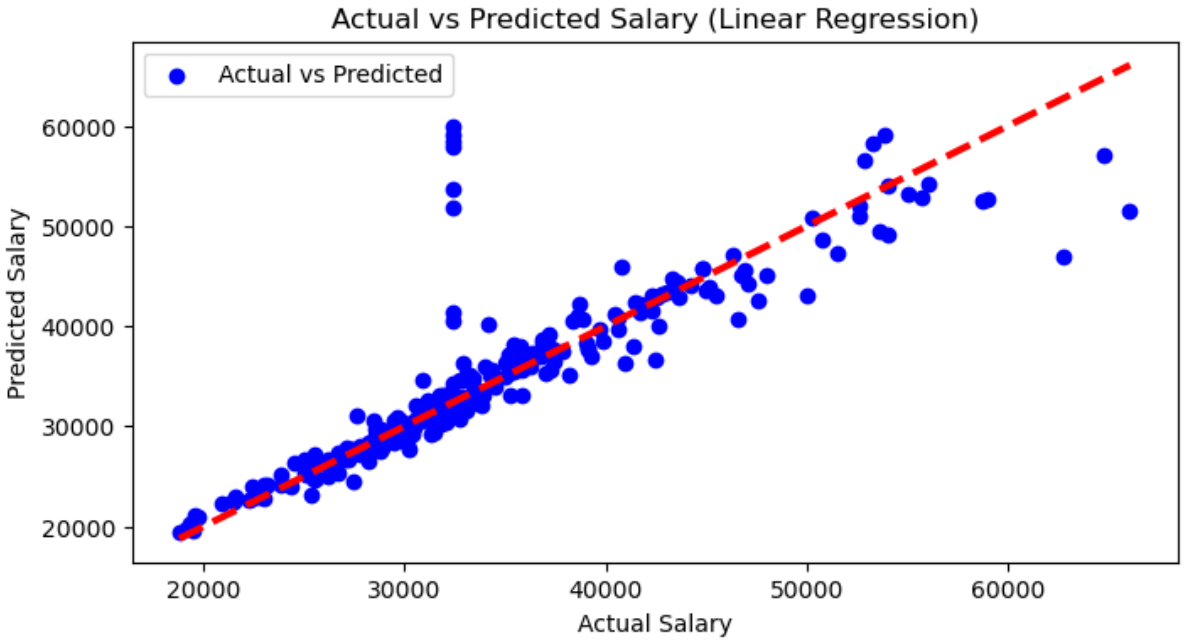


Figure 8: ML Actual vs Predicted mean Salary

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

2. Ridge and Lasso

Ridge regression is a variant of linear regression that incorporates regularization to address potential issues with multicollinearity and overfitting. Like multilinear regression, it seeks to model the relationship between multiple independent variables and a dependent variable. However, ridge regression introduces a regularization term to the traditional least squares objective function.

Ridge keeping the model coefficients from growing too large. It achieves this balance by penalizing the sum of squared coefficients, effectively shrinking them towards zero. This regularization term, controlled by a hyperparameter called lambda (λ), helps prevent overfitting by discouraging overly complex models with large coefficient values.

It is mitigating multicollinearity, where predictors are highly correlated, ridge regression improves the stability of coefficient estimates. This is particularly valuable when dealing with datasets containing correlated predictors, as it helps avoid inflated coefficient estimates and reduces the sensitivity of the model to small changes in the data.

Its ability to handle multicollinearity and prevent overfitting makes it a valuable choice for many regression tasks, particularly when dealing with correlated predictors.

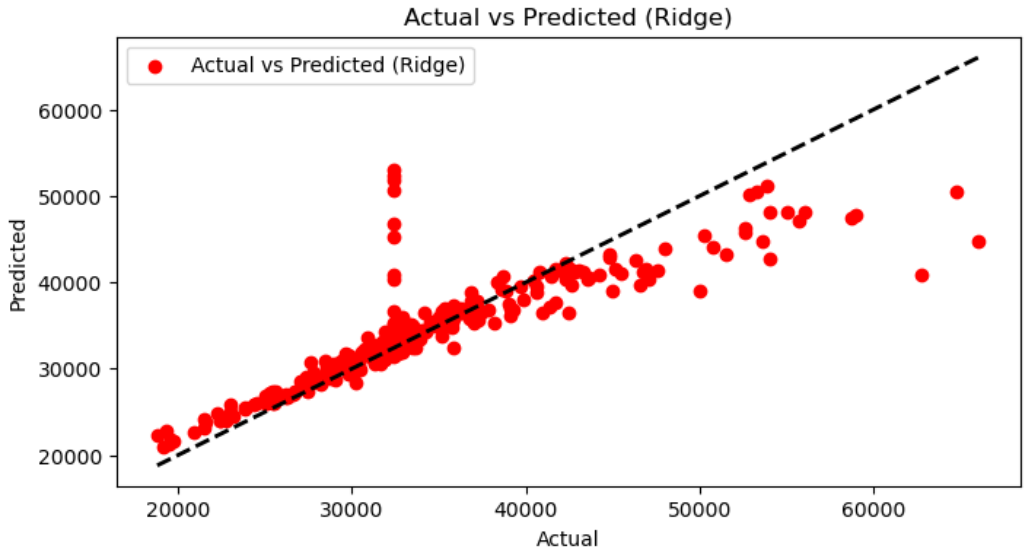


Figure 9: ridge actual vs predicted

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

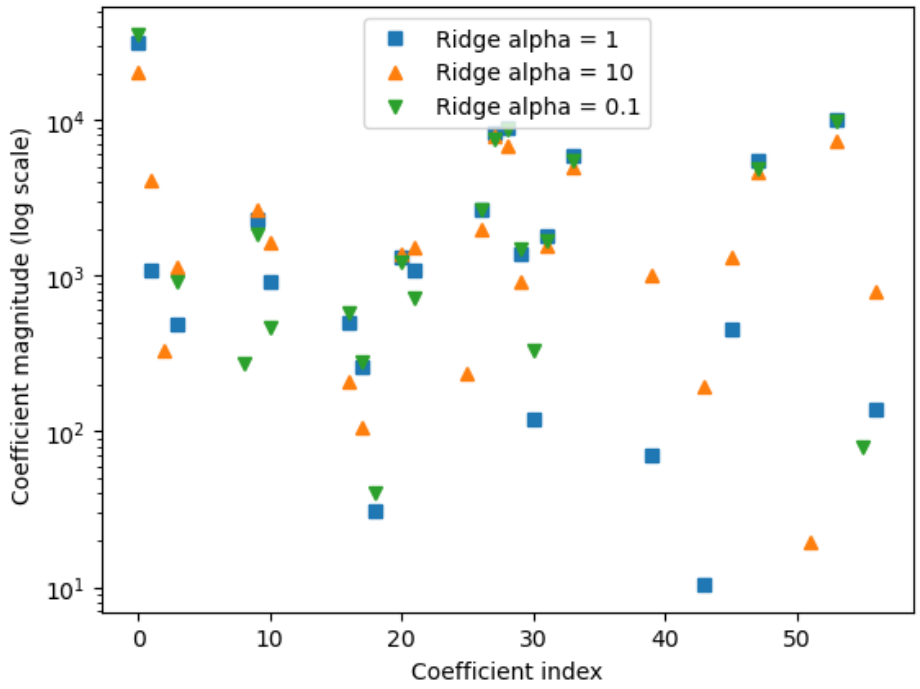


Figure 10 Ridge Coeffcient magnitude

RandomForest

Random Forest Regression is an ensemble learning method used for regression tasks. It builds multiple decision trees during training and outputs the average prediction of the individual trees, which improves the predictive accuracy and controls overfitting. The method is robust to noise and can handle a large number of input variables, making it suitable for a wide range of applications.

Key Parameters of Random Forest Regression

n_estimators: The number of trees in the forest, increasing the number of trees generally improves the model's performance but also increases computational cost.

max_depth: The maximum depth of each tree, limiting the depth of the trees prevents overfitting. A deeper tree might capture more details but can also overfit the training data, whereas a shallower tree might be less complex and generalize better.

min_samples_split: The minimum number of samples required to split an internal node, a higher value prevents the model from learning overly specific patterns, reducing overfitting. However, too high a value might cause underfitting.

min_samples_leaf: The minimum number of samples required to be at a leaf node, controls the minimum size of the leaf nodes. Setting this to a higher value ensures that leaf nodes contain more samples, which can help in smoothing the model and reducing overfitting.

I perform a grid search for hyperparameter tuning and evaluate the performance of the best RandomForestRegressor model, using cross-validation to evaluate the performance of each combination of parameters, and selecting the combination that give the best results.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

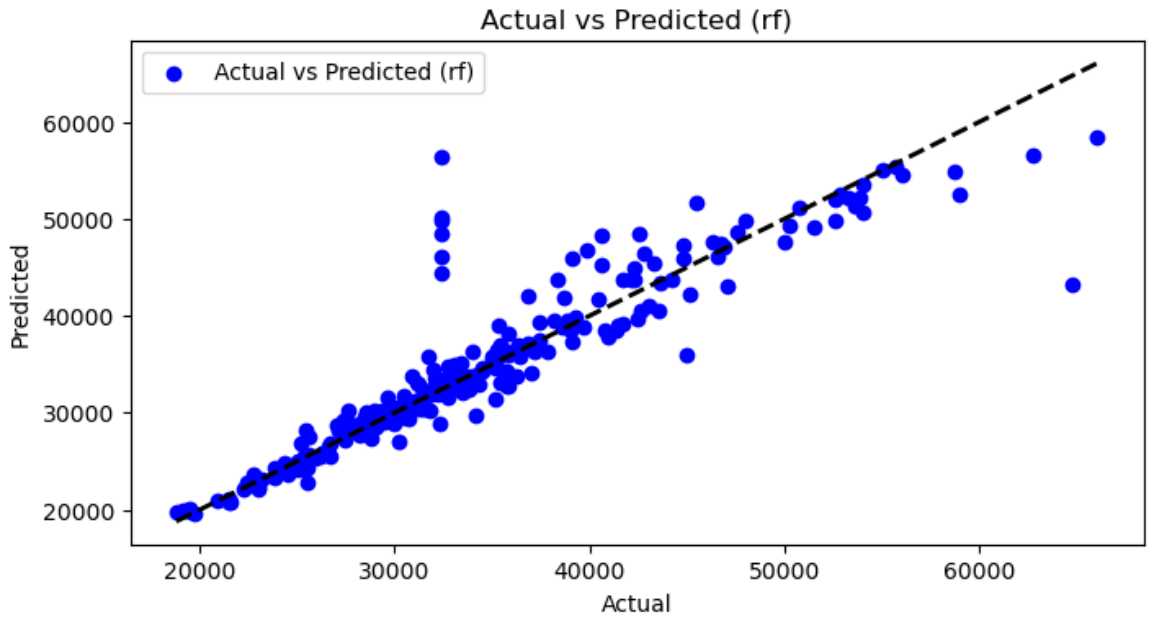


Figure 11: Actual vs Predected salary rf

K-Nearest Neighbors Regression (KNN Regression)

KNN regression is a simple, instance-based learning method. It predicts the value of a target variable by averaging the values of its k-nearest neighbors.

K-Nearest Neighbors (KNN) regressor is a simple, yet effective, instance-based learning algorithm used for regression tasks. It predicts the value of a target variable by identifying the k-nearest data points in the feature space and averaging their target values. The choice of k, the number of neighbors, is a crucial hyperparameter that affects the model's bias-variance tradeoff: a smaller k makes the model more sensitive to noise (high variance), while a larger k smooths the predictions, potentially leading to underfitting (high bias). So we plot k against Rsqr to choose the optimal number for K

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

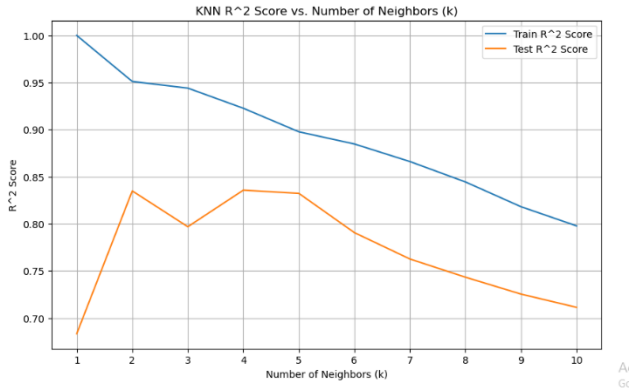


Figure 12: #K vs R^2

KNN regression is non-parametric, meaning it makes no assumptions about the underlying data distribution, making it flexible for various types of datasets. However, it can be computationally intensive, especially with large datasets, as it requires distance calculations for each prediction. Additionally, the performance of KNN regression can degrade with high-dimensional data due to the curse of dimensionality

```
KNN Train RMSE: 2280.5866
KNN Test RMSE: 3472.5225
KNN Train R^2 Score: 0.9227
KNN Test R^2 Score: 0.8357
KNeighbors CV Score: 0.6200
```

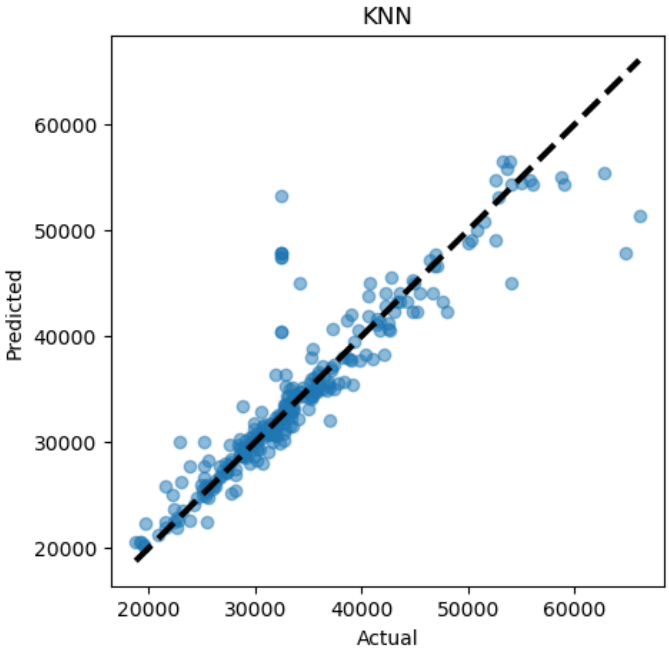


Figure 13: KNN Actual vs Predicted mean salary

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

Support Vector Machine (SVM) regressor, is a powerful and versatile machine learning algorithm used for regression tasks. SVR finds a hyperplane in a high-dimensional space that best fits the data while allowing for some flexibility through a margin of tolerance defined by the epsilon parameter. The key hyperparameters in SVR include the regularization parameter C, which balances the trade-off between achieving a low error on the training data and maintaining a smooth, less complex model, and the kernel function, which transforms the input data into a higher-dimensional space to handle non-linear relationships. Common kernel choices include linear, polynomial, and radial basis function (RBF), each offering different advantages depending on the data's characteristics. SVR is particularly effective in high-dimensional spaces and is robust to overfitting. However, it requires careful tuning of hyperparameters. I used Grid search optimize the hyperparameters of a (SVM) regressor, ensuring the best performance for a given dataset. such as param_grid, 'epsilon', 'kernel', 'gamma', 'degree': [2, 3, 4] and 'coef0'. the grid search systematically evaluates all possible combinations of these hyperparameters.

The C parameter controls the trade-off between a smooth decision boundary and classifying training points correctly

epsilon defines the margin of tolerance for errors.

The kernel parameter specifies the type of kernel to be used, with options like linear, polynomial, RBF, and sigmoid

The gamma parameter influences the shape of the decision boundary in RBF, polynomial, and sigmoid kernels

coef0 represents the independent term in the kernel function.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

```
Best parameters found:  {'C': 100, 'coef0': 1.0, 'degree': 4, 'epsilon': 1, 'gamma': 'scale', 'kernel': 'poly'}

Best SVM Train RMSE: 3220.235
Best SVM Test RMSE: 4446.4943
Best SVM Train R^2 Score: 0.8458
Best SVM Test R^2 Score: 0.7306
KNeighbors CV Score: 0.5910
```

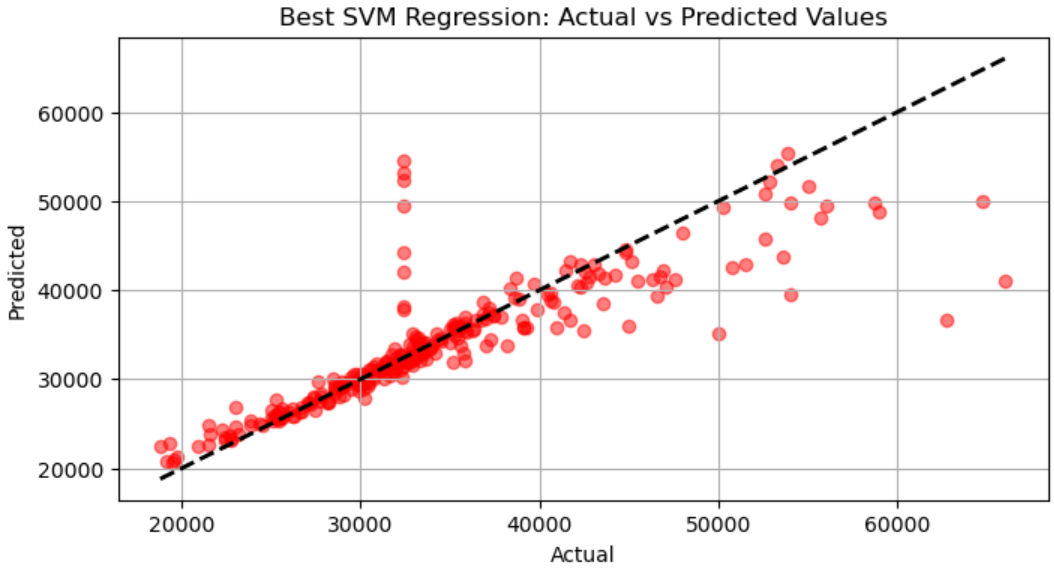


Figure 14: SVM Actual vs predicted mean salary

Model evaluation

RMSE

Root Mean Square Error (RMSE) is a widely used metric in the field of machine learning and statistics to evaluate the performance of regression models. It provides a measure of how well the model's predictions match the actual values. The RMSE is calculated by taking the square root of the average of the squared differences between the predicted values and the actual values.

$$RMSE = \sqrt{\frac{\Sigma_i^N (x_i - \hat{x}_i)^2}{N}}$$

Figure 15 the RMSE for each model Random forest is the best performance model it has the lowest RMSE value.

R-squared (R^2), also known as the coefficient of determination, is a statistical measure used in regression analysis to assess the goodness of fit of a model to the data. It quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total variance}}$$

$$R^2 = (R)^2 = 1 - \frac{SSE}{SS_{yy}}$$

R-squared ranges from 0 to 1, where 1 indicates a perfect fit, meaning all variability in the dependent variable is explained by the independent variables, while 0 indicates no linear relationship between the dependent and independent variables. R-squared is a valuable tool for evaluating the predictive power and overall adequacy of a regression model.

Figure 16 the Rsqr for each model, Random Forest is the best performance model it has the highest Rsqr value.

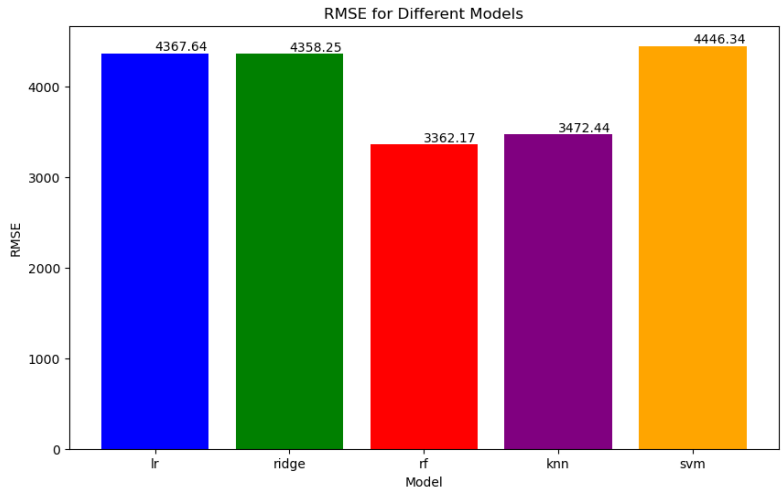


Figure 15: RMSE for each model

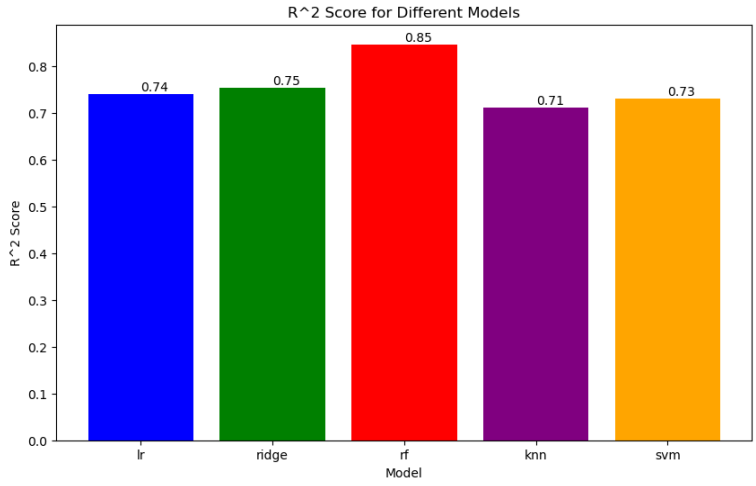


*Figure 16: R*2 for each model*

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

Evaluating machine learning models extends beyond merely assessing accuracy; it requires a comprehensive analysis that includes stability, financial cost. Stability ensures that the model's performance is consistent across different datasets and over time, which is crucial for reliability in real-world applications.

Using cross-validation provides a robust measure of a model's performance by evaluating it on multiple subsets of the data, reducing the likelihood of overfitting. This technique ensures that the model's accuracy is not dependent on a particular split of the data, offering a more reliable estimate of its generalization ability.

In Figure 17,cross validation of 5 folds for each model. linear regression after PCA and random forrest are the top classifiers.

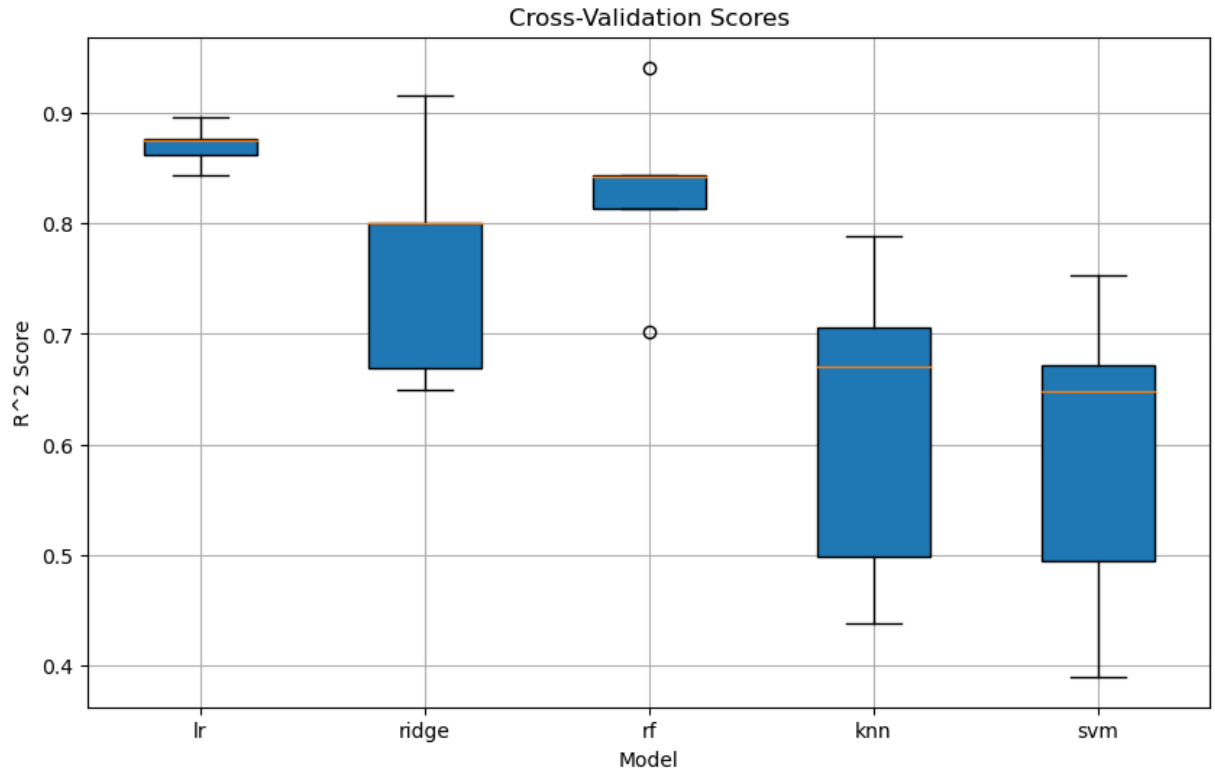


Figure 17: Cross Validation Scores for each model

When comparing different regression models such as multilinear regression, ridge regression, random forest regressor, k-nearest neighbors (KNN) regressor, and support vector machine (SVM) regressor in terms of cost, it's important to consider several factors that contribute to the overall cost. These factors include computational resources, model training time, data requirements, and maintenance costs.

https://github.com/hh2-radwan/-Machine-Learning-and-Predictive-Analytics---Coursework.git

1. Multilinear Regression is cost-effective in terms of both computational resources and maintenance. It's suitable for problems where relationships between variables are linear.

2. Ridge Regression incurs slightly higher costs than multilinear regression due to regularization but remains cost-effective for large datasets.

3. Random Forest Regressor is computationally expensive but provide high accuracy and robustness, making them suitable for complex, large-scale problems.

4. K-Nearest Neighbors (KNN) Regressor has low training costs but high prediction costs, making it less suitable for real-time predictions or very large datasets.

5. Support Vector Machine (SVM) Regressor are computationally expensive and require significant resources and expertise to tune and maintain, but they can offer high performance for specific applications.

The choice of model should balance financial cost with the desired accuracy . Multilinear or Ridge Regression would be best models for this dataset.

Comments

The dataset selected was a poor choice. Independent variables have high multicollinearity. This issue significantly impacted the model's performance, also Unusual high RMSE values.