**Task 3: Principal Component Analysis**

Joseph Emmanuel Cayetano

Western Governors University

D600: Statistical Data Mining Task 3

November 25, 2024

1. **Create your subgroup and project in GitLab**

**GitLab repository URL:** https://gitlab.com/wgu-gitlab-environment/student-repos/jcayet5/d600-statistical-data-mining/-/tree/task3branch?ref\_type=heads

**Repository branch history screenshot:**

A screenshot of a computer

Description automatically generated

**B1. Propose one research question that is relevant to a real-world organization situation captured in the dataset that can be answered using linear regression**

The research question that is relevant to the housing information dataset and can be answered using linear regression is: Which property characteristics most significantly determine housing prices?

**B2. Define one goal of the data analysis**

One goal of the data analysis is to identify the key property attributes that significantly influence housing prices, using linear regression. This goal can help stakeholders such as real estate companies and developers prioritize which features to include in new construction projects that have the highest potential return and develop accurate house pricing strategies. This data analysis goal is within the scope of the scenario and data.

**C1. Explain how PCA can be used to prepare the selected dataset for regression analysis. Include expected outcomes**

To perform PCA on the housing dataset, the first step is to standardize the data using the StandardScaler() function from the sklearn library to ensure all continuous variables are on the same scale. The second step is to transform the standardized data into principal components, which are linear combinations of the original variables in the dataset. Principal components are ordered by the amount of variance they explain. For example, the first principal component (PC1) captures the most variance, followed by the second (PC2), and so on. The third step is choosing the number of principal components. The key is choosing enough PCAs that capture most variance. The last step is to use the chosen PCAs as input variables for the regression model instead of the original variables.

By performing PCA on the regression model, there are a few expected outcomes. First, it simplifies the regression model by reducing the number of variables while maintaining most information in the data. Second, it reduces multicollinearity between variables since PCA creates uncorrelated components from correlated features. Third, it improves the regression model's performance by focusing only on components that capture the most variance.

**C2. Summarize one assumption of PCA**

One assumption of PCA is that it assumes a linear relationship between variables. PCA performs well when variables show straight-line relationships, as it can then effectively reduce the number of variables and remove multicollinearity while keeping the most important patterns in the data. If the relationships between variables are non-linear, PCA may not capture the important patterns in the data. Data transformations must be performed to turn non-linear variables into linear, making it suitable for PCA.

**D1. Identify the continuous dataset variables that you will need to answer the research question proposed in part B1**

I have identified the continuous variables that I will use to answer the research question from part B1. Price will be the dependent variable because our research question focuses on determining which property features most significantly impact housing prices. Price represents the housing prices we want to predict and understand, making it the logical choice as the dependent variable. The other continuous variables that I have chosen are: SquareFootage, NumBathrooms, NumBedrooms, BackyardSpace, CrimeRate, SchoolRating, AgeOfHome, DistanceToCityCenter, EmploymentRate, PropertyTaxRate, RenovationQuality, LocalAmenities, TransportAccess, and PreviousSalePrice. These variables will be the independent variables for our regression model. For SquareFootage, larger properties typically have higher prices. For NumBathrooms, properties with multiple bathrooms are typically more expensive. For NumBedrooms, properties with multiple bedrooms generally command higher prices. For BackyardSpace, houses with larger backyards tend to have higher prices. For CrimeRate, properties in areas with lower crime rates are typically associated with higher property prices. For SchoolRating, properties in areas with good school systems are generally more expensive. For AgeOfHome, newer homes typically have higher prices than older homes because they were built more recently. For DistanceToCityCenter, houses that are located closer to urban centers are more desirable. For EmploymentRate, properties in areas with higher employment rates tend to be more expensive. For PropertyTaxRate, higher property taxes might impact home buyers' decisions, potentially influencing the price. For RenovationQuality, homes that are renovated well generally command higher prices. For LocalAmenities, properties that have many nearby amenities are typically linked to higher property values. For TransportAccess, convenient transport connections can increase a property's appeal. For PreviousSalePrice, it tells historical value trends, helping model price growth or depreciation. I selected these variables as independent variables because they represent physical attributes of a property, the surrounding neighborhood, and accessibility. These factors are likely to influence property prices and are well-suited for analysis using linear regression.

**D2. Standardize the continuous dataset variables identified in part D1. Include a copy of the cleaned dataset**

A screenshot of a computer

Description automatically generated

A screenshot of a calculator

Description automatically generated

In the code, I stored the dependent variable Price into target\_variable, and stored the 14 independent variables into independent\_variables. I created a feature subset containing only the independent variables and stored them into the features variable. Then, I initialized an instance of StandardScaler from sklearn library, and fit the scaler to the features and transformed the data so each feature has a mean of 0 and a standard deviation of 1. The standardized\_df variable is a dataframe that contains the standardized features. The standardized\_df\_with\_dependent variable is a dataframe that contains the standardized features and the dependent variable Price.

Looking at the standardized data, all the independent variables have been properly transformed. Their mean is effectively 0, and their standard deviation is approximately 1. All variables in the data are now on the same scale, making them suitable for PCA.

A close up of a text

Description automatically generated

Saving the standardized dataframe as a CSV file for submission.

**D3. Describe the dependent variable and all independent variables from part D1 using descriptive statistics, including a screenshot for each of these variables**

A screenshot of a computer

Description automatically generated

The Price variable has a total count of 7000, mean value of 307282, minimum value of 85000, maximum value of 1046676, and a range value of 961676 (max-min). Price also has many mode values, as indicated by the results from using the mode() function.

A screenshot of a computer program

Description automatically generated

The SquareFootage variable has a total count of 7000, mean value of 1048.94, mode value of 550, minimum value of 550, maximum value of 2844.70, and a range value of 2324 (max-min).

A screenshot of a computer

Description automatically generated

The NumBathrooms variable has a total count of 7000, mean value of 2.1, mode value of 1.0, minimum value of 1, maximum value of 5.8, and a range value of 4.8 (max-min).

A screenshot of a computer

Description automatically generated

The NumBedrooms variable has a total count of 7000, mean value of 3.0, mode value of 3, minimum value of 1.0, maximum value of 7, and a range value of 6 (max-min).

A screenshot of a computer

Description automatically generated

The BackyardSpace variable has a total count of 7000, mean value of 511.5, mode values of 300.08, 418.29, and 516.29, minimum value of 0.39, maximum value of 1631, and a range value of 1630.97 (max-min).

A screenshot of a computer

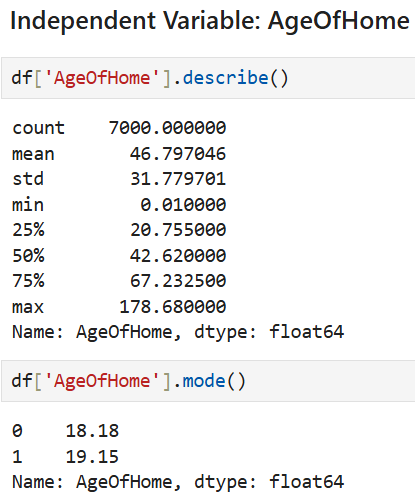
Description automatically generated

The CrimeRate variable has a total count of 7000, mean value of 31.2, mode value of 34.01, minimum value of 0.03, maximum value of 99.73, and a range value of 99.70 (max-min).

A screenshot of a computer

Description automatically generated

The SchoolRating variable has a total count of 7000, mean value of 6.94, mode value of 10.0, minimum value of 0.22, maximum value of 10, and a range value of 9.78 (max-min).



The AgeOfHome variable has a total count of 7000, mean value of 46.79, mode values of 18.18 and 19.15, minimum value of 0.01, maximum value of 178.68, and a range value of 178.67 (max-min).

A screenshot of a computer

Description automatically generated

The DistanceToCityCenter variable has a total count of 7000, mean value of 17.47, mode value of 8.29, minimum value of 0, maximum value of 65.20, and a range value of 65.20 (max-min).

A screenshot of a computer

Description automatically generated

The EmploymentRate variable has a total count of 7000, mean value of 93.71, mode value of 99.9, minimum value of 72.05, maximum value of 99.90, and a range value of 27.85 (max-min).

A screenshot of a computer

Description automatically generated

The PropertyTaxRate variable has a total count of 7000, mean value of 1.50, mode value of 1.43, minimum value of 0.01, maximum value of 3.36, and a range value of 3.35 (max-min).

A screenshot of a computer program

Description automatically generated

The RenovationQuality variable has a total count of 7000, mean value of 5.00, mode value of 10.0, minimum value of 0.01, maximum value of 10.0, and a range value of 9.99 (max-min).

A screenshot of a computer

Description automatically generated

The LocalAmenities variable has a total count of 7000, mean value of 5.93, mode value of 10.0, minimum value of 0, maximum value of 10, and a range value of 10 (max-min).

A screenshot of a computer

Description automatically generated

The TransportAccess variable has a total count of 7000, mean value of 5.98, mode value of 10.0, minimum value of 0.01, maximum value of 10, and a range value of 9.99 (max-min).

A screenshot of a computer

Description automatically generated

The PreviousSalePrice variable has a total count of 7000, mean value of 284509.40, minimum value of -8356.90, maximum value of 1296607, and a range value of 1304963.90 (max-min). PreviousSalePrice has also many mode values, as indicated by the results from using the mode() function.

**E1. Determine the matrix of all the principal components ( REVISION FIXED)**

A screenshot of a computer

Description automatically generated

In the code, I created a PCA object and used that object to fit on the standardized data and transform the data into principal components. Then, I created a dataframe that contains the loadings of the 14 independent variables for each principal component.

In the loadings matrix, each column represents one principal component. There are 14 principal components in the matrix, which equals the number of the original independent variables. Each cell in the matrix indicates how much each independent variable influences each principal component. In PC1, independent variables such as SquareFootage, SchoolRating, RenovationQuality, and PreviousSalePrice have the highest positive contributions, indicating they heavily influence PC1. Since PC1 captures the most variance in the dataset, organizations should take note of the aforementioned variables for investment or improvement regarding properties.

**E2. Identify the total number of principal components (that should be retained), using the elbow rule or the Kaiser rule. Include a screenshot of the scree plot**

A graph with blue dots and lines

Description automatically generated

In this code, I extracted the explained variance ratio from the PCA object to determine how much variance each principal component explains. Then, I created a scree plot to visualize the explained variance for each principal component so we can use the elbow rule. The elbow rule is a method used to identify the number of principal components to keep without being overly complex.

In the scree plot, the most noticeable elbow occurs at Principal Component 2 (PC2), however since we aim to retain at least 80% of the total variance for our regression model, we will not be using PC2. A less distinct elbow appears at PC10, where there is a slight change in the slope, and the explained variance begins to flatten further. Selecting PC10 would be ideal for our regression model because PC10 retains approximately 87% of the total variance and provides a balance between reducing dimensionality and retaining most of the critical information in the housing dataset, ensuring that significant patterns are captured while minimizing complexity. Therefore, the first 10 principal components will be retained.

**E3. Identify the variance of each of the principal components identified in part E2**

**A screenshot of a computer program

Description automatically generated**

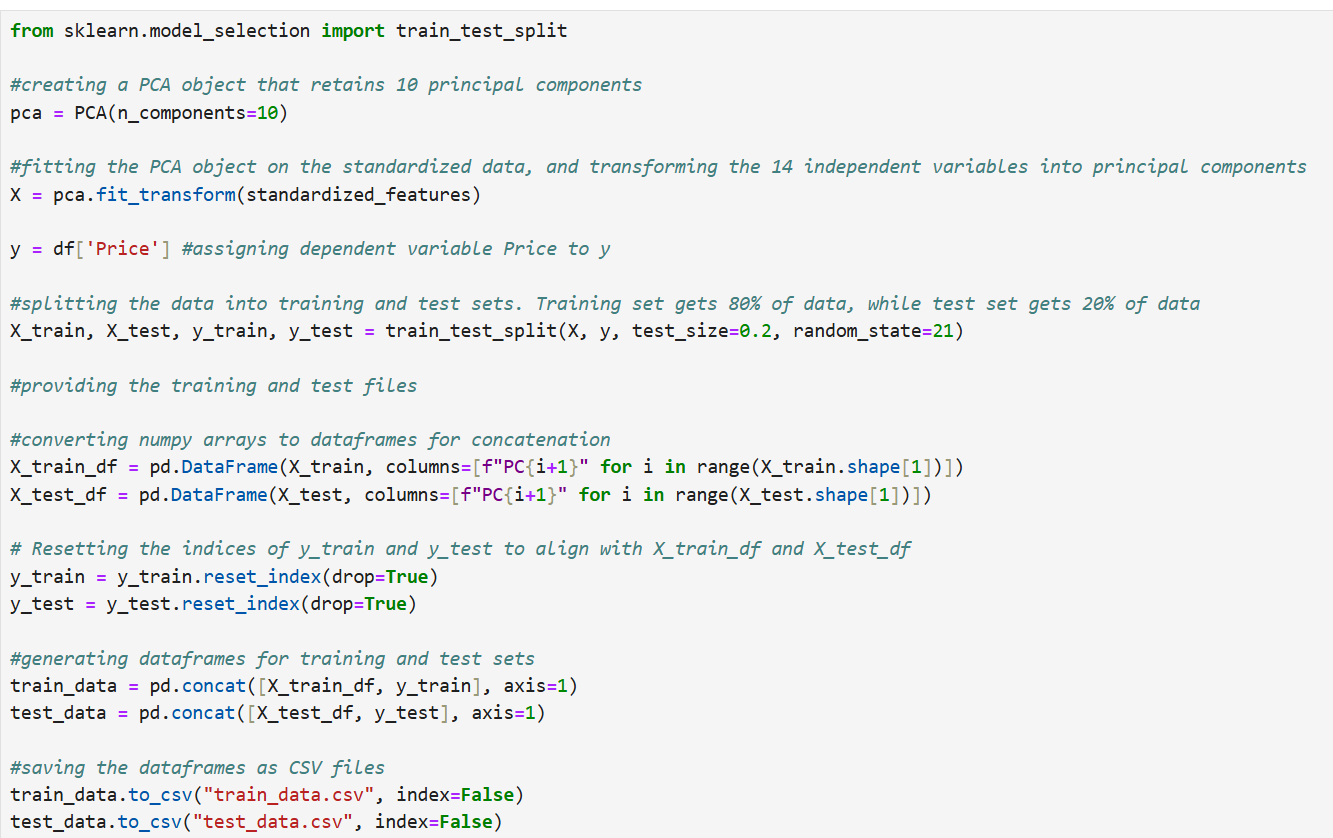
In this code, I created a loop that iterates over the first 10 principal components, outputting each component's individual variance explained. I formatted the output values to 4 decimal places.

PC1 explains the highest variance at 23.10%. PC2 explains 9.69% of variance. PC3 explains 8.09% of variance, PC4 explains 7.61% of variance. PC5 explains 7.22% of variance. PC6 explains 6.68% of variance. PC7 explains 6.56% of variance. PC8 explains 6.45% of variance. PC9 explains 6.04% of variance. PC10 explains 5.92% of variance. Overall, the first 10 principal components explain 87.37% of total variance, exceeding our minimum threshold of 80% for regression modeling.

**E4. Summarize the results of your PCA (Revision Fixed)**

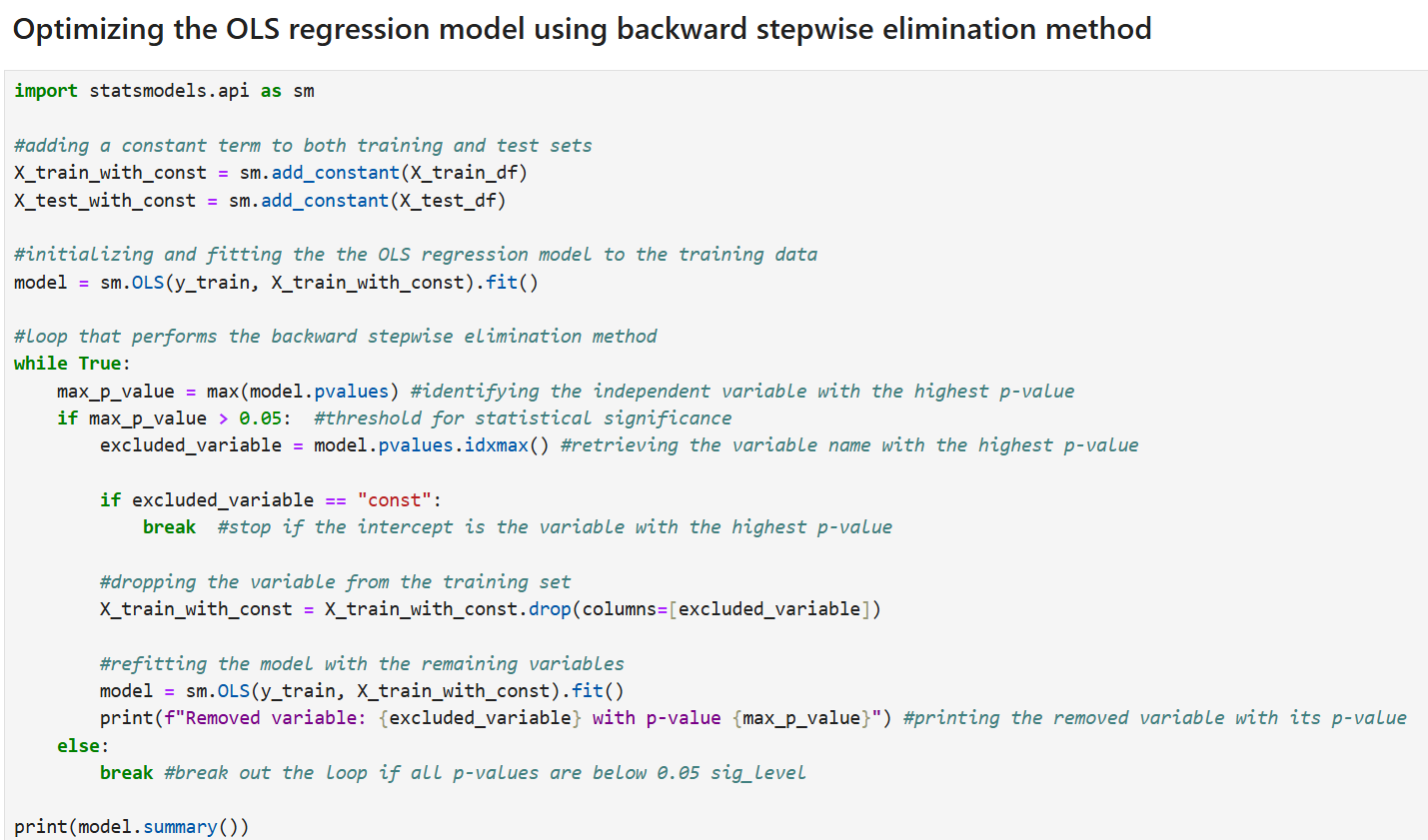
In part E1, we created a loading matrix of the 14 independent variables for each principal component. In PC1, independent variables such as SquareFootage, SchoolRating, RenovationQuality, and PreviousSalePrice have the highest positive contributions, indicating they heavily influence PC1. Since PC1 captures the most variance in the dataset, organizations should take note of the aforementioned variables for investment or improvement regarding properties. In part E2, we created a scree plot. By using the elbow rule in the scree plot, we identified the total number of principal components to retain, which is 10. We selected the first 10 principal components, with PC1 accounting for nearly a quarter of the total variance, while the remaining components contribute progressively less. Together, these 10 components explain 87% of total variance, which is ideal for regression modeling because it means that the most important patterns in the housing dataset are captured, enabling the regression model to effectively make predictions with the dataset. The selected components also reduced the dimensions of the housing dataset from 14 variables to 10 principal components while preserving most of the information. This simplifies the dataset because only the important information is retained while the irrelevant features are removed, making it less likely for the regression model to overfit to noise in the dataset. Additionally, the reduced dataset makes it easier to understand the key patterns that significantly impact housing prices, without being overwhelmed by all 14 original variables. We will be using the reduced housing dataset that includes the 10 principal components throughout the regression analysis.

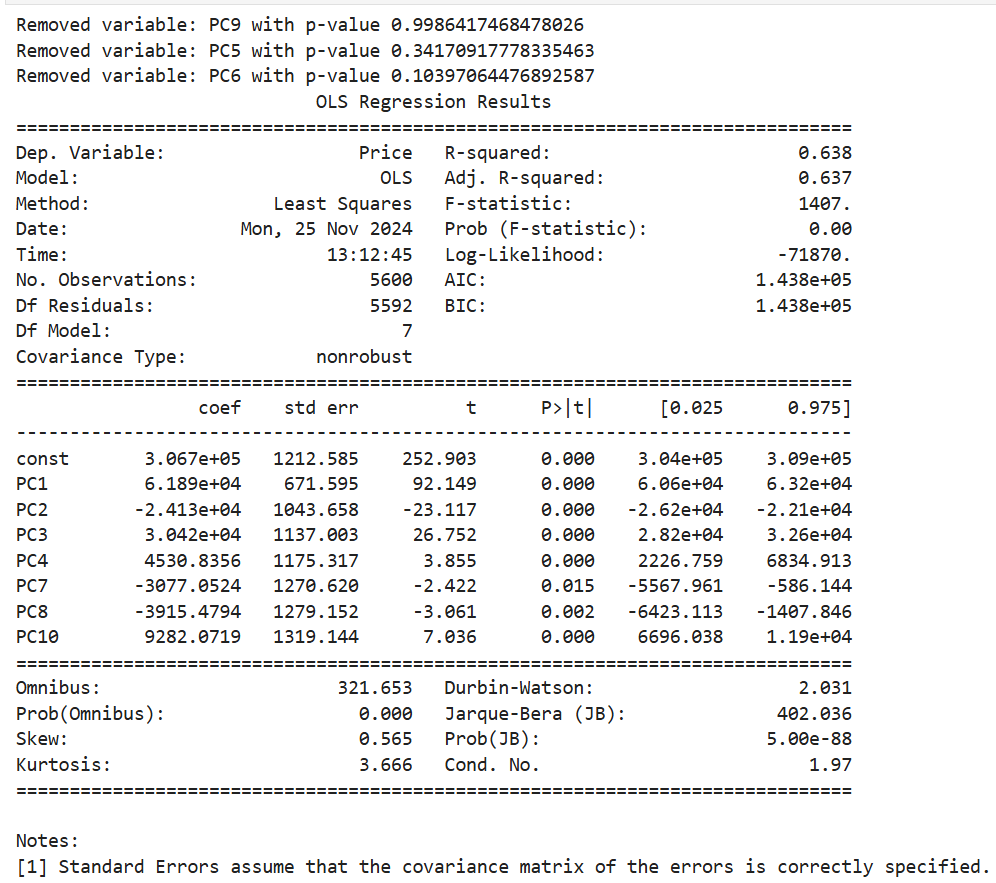
**F1. Split the data into two datasets, with a larger percentage assigned to the training dataset and a smaller percentage assigned to the test dataset. Provide the files**



In this code, I used the PCA() function to create a PCA object that is configured to retain 10 principal components. Then, I fit the PCA model to the standardized features and transformed the 14 independent variables into principal components. The 10 principal components are stored in the X variable, while the dependent variable Price is stored in the y variable. I used the train\_test\_split() function from sklearn library to split the housing dataset into training and test sets. The training set receives 80% of the data, while the test set receives 20% of the data. Next, I converted the two NumPy arrays X\_train and X\_test into pandas dataframes with each column labeled as PC1 through PC10 for better interpretability. I reset the indices of y\_train and y\_test using the reset\_index() function to ensure alignment with X\_train\_df and X\_test\_df. Then, I merged X\_train\_df with y\_train to create a single dataframe for the training set, and merged X\_test\_df with y\_test to create a single dataframe for the test set. I saved both training and test dataframes as CSV files for submission.

**F2. Use the training dataset to create and perform a regression model. Optimize the regression model and provide a screenshot of the summary of the optimized model or the model parameters**





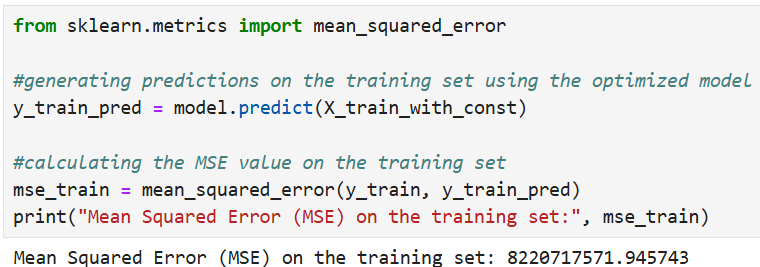
In this code, I optimized the Ordinary Least Squares (OLS) regression model by performing the backward stepwise elimination method. The backward stepwise elimination method simplifies the model and improves its performance by including all the independent variables in the model and iteratively removing the least significant variables with p-values above 0.05, until all remaining variables are statistically significant. Before optimizing the model, I used the add\_constant() function from statsmodels library to add a constant term to both training and test sets. Then, I initialized and fit the OLS regression model to the training data. I created a while loop that performs the backward stepwise elimination method. Inside the loop, the function pvalues retrieves the independent variable with the highest p-value. If the variable's p-value is greater than the significance level (0.05), the variable gets removed from the training dataset. After the removal process, the model gets refitted using the updated training dataset. The loop continues until all the remaining variables have p-values below 0.05, indicating that all variables are statistically significant. After the loop completes, the optimized regression model's summary is displayed using the summary() function.

We'll go over several key metrics from this OLS regression results. The R-squared value of the regression model is 0.638, indicating that 63.8% of the variance in housing prices is explained by the 7 remaining principal components. This R-squared value suggests that there is a fairly strong relationship between the principal components and the dependent variable Price. The adjusted R-squared value of the model is 0.637 and it is slightly lower than the R-squared value, which means the model is not overfitting and does not include unnecessary principal components because the adjusted R-squared value did not get penalized as much. This confirms the effectiveness of applying the backward stepwise elimination method to the regression model, as all of the principal components that were retained in the model are efficient and relevant for predicting house prices.

The F-statistic value of the regression model is 1407, indicating that the overall model is statistically significant and provides a much better fit to the data compared to a model without any predictors. The Prob (F-statistic) value is 0.00, which means the model is statistically significant as a whole and its principal components have a strong relationship with the dependent variable Price.

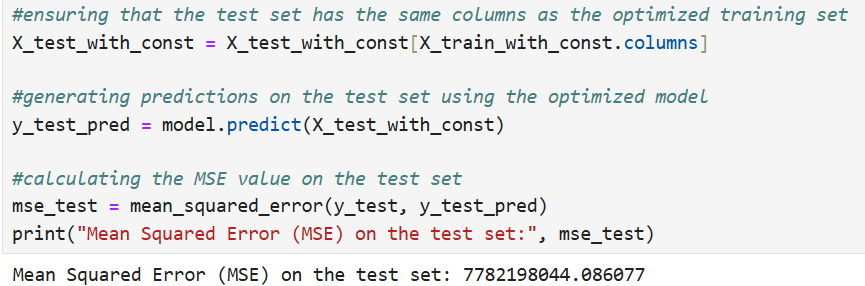
Regarding coefficient estimates, the principal components with a positive coefficient are PC1, PC3, PC4, and PC10. These components increase house prices when they increase. The principal components with a negative coefficient are PC2, PC7, and PC8. These components decrease house prices when they increase. As for the p-values, PC1, PC2, PC3, PC4, PC7, PC8, and PC10 are all statistically significant predictors of housing prices, as their p-values are below 0.05. These retained principal components play a key role in determining variations in housing prices. The variables PC9, PC5, and PC6 were removed from the model using the backward stepwise elimination method due to their p-values exceeding 0.05, indicating they were not statistically significant.

**F3. Give the mean squared error (MSE) of the optimized model used on the training set**



In this code, I used the optimized regression model to generate predictions on the training dataset, X\_train\_with\_const. The predicted values for the target variable, Price, were stored in the y\_train\_pred variable. Then, I calculated the average squared difference between the predicted and actual values by using the mean\_squared\_error() function from sklearn. The Mean Squared Error (MSE) on the training set is 8220717571.94, which represents the average squared difference between predicted and actual house prices. This large MSE value should be expected because the house prices in the dataset are typically in the range of hundreds of thousands. This MSE value also means that the model makes moderately accurate predictions.

**F4. Run the prediction on the test dataset using the optimized regression model to give the accuracy of the prediction model based on the mean squared error (MSE)**



In this code, I started by ensuring that the columns in the test set were aligned with the columns in the optimized training set to get consistent results. I used the optimized regression model to make predictions on the test set, X\_test\_with\_const. The model's predicted values are stored in the y\_test\_pred variable. Then, I compared the predicted prices with actual test prices and calculated their average squared difference by using the mean\_squared\_error() function from sklearn. The Mean Squared Error (MSE) on the test set is 7782198044.08, which represents the average squared difference between the actual and predicted house prices for the test dataset. The minimal difference between the training and test set MSE values suggests that the optimized regression model performs equally well on both the training data and new data. This also means that the model achieves a balance between underfitting and overfitting, making it well-suited for predicting house prices with acceptable accuracy.

**G1. List the packages or libraries you have chosen for Python and justify them**

The libraries that I have used throughout the regression analysis are pandas, NumPy, matplotlib, scikit-learn, and statsmodels. I used the pandas library primarily for data manipulation and analysis. For example, the read\_csv() function from pandas allowed me to read and load the housing dataset into a pandas dataframe. Another example is the describe() function from pandas, and the function retrieved the descriptive statistics for each variable in the data. The to\_csv() function is also another helpful function from pandas, as it was responsible for saving both training and test sets as CSV files so I could use them for submission. The NumPy library is a valuable library for numerical operations. Two of the functions from the NumPy library that I used in the analysis were the array and shape functions. The array function helped me convert the PCA-transformed data into NumPy arrays for compatibility with the scikit-learn functions. As for the shape function, it allowed me to check the dimensions of both the training and test datasets. Matplotlib is also another valuable library in the regression analysis, as it helped me generate the scree plot that was used to determine the optimal number of components for PCA. Matplotlib also allowed me to customize the title and the labels for the scree plot to make it look presentable. The scikit-learn library is a useful library for machine learning. The train\_test\_split() function from scikit-learn allowed me to split the housing dataset into training and test sets for regression modeling. Also, the standardscaler() function from scikit-learn was responsible for standardizing the continuous variables in the dataset to ensure compatibility for PCA. Additionally, scikit-learn contains functions that evaluate model performance like the mean\_squared\_error() function, which was used in the analysis to examine the performance of the optimized regression model on both the training and test sets. I used the statsmodels library to build a statistical model. The OLS() function from statsmodels created the regression model that was optimized during the backward stepwise elimination process, and provided model summaries that include coefficients, p-values, and R-squared.

**G2. Discuss the method used to optimize the model and justification for the approach**

The backward stepwise elimination method was the method used to optimize the regression model. This method "begins with a model that contains all variables then starts removing the least significant variables one after the other until a pre-specified stopping rule is reached" (Chouiery, 2019, par. 12). This method generally improves the model's performance by eliminating unnecessary predictors that do not significantly impact the target variable. In our data analysis, I built a while-loop that performs the backward stepwise elimination method. The method starts by fitting the regression model to the training data, which includes all 10 principal components. Then, the pvalues function retrieves the principal component with the highest p-value and checks whether the p-value exceeds 0.05. If it does, the principal component gets removed from the training dataset. After the removal process, the regression model gets refitted using the updated training dataset. The process repeats until all remaining components have p-values less than 0.05. The removed components were PC9, PC5, and PC6 since they had p-values exceeding 0.05, which indicates that they did not significantly impact housing price prediction. The components that were retained are PC1, PC2, PC3, PC4, PC7, PC8 and PC10 since they have p-values less than 0.05, indicating that they contribute meaningfully to predicting housing prices.

The justification for applying the backward stepwise elimination method to the regression model is to reduce the complexity of the model by eliminating principal components that do not have significant influence on predicting house prices. This makes the model simpler and easier to interpret. The optimization method also improves the model's performance by lowering its risk of overfitting since the model focuses solely on the significant components that impact housing prices. This results in improved generalization to unseen data.

**G3. Discuss the verification of assumptions used to create the optimized model**

A screen shot of a graph

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

There are three assumptions that I have verified for the optimized regression model to confirm its accuracy and validity for predicting housing prices. The first assumption is linearity where the relationship between the principal components and the target variable Price should be linear. I created a residual plot to verify the linearity assumption in the model. The residuals were calculated by subtracting the predicted house prices from the actual house prices. In the residual plot, the residuals are distributed randomly around the red center line of zero, which indicates that the model effectively represents the linear relationship between the principal components and Price. There are also no distinct patterns in the residuals, suggesting that the model satisfied the linearity assumption. The second assumption is independence of residuals where residuals should not be correlated with each other. To verify that the residuals in the model are independent, I used the Durbin-Watson test, which checks for autocorrelation in the residuals. A Durbin-Watson statistic value of 2 implies no autocorrelation, whereas values above or below 2 imply negative or positive correlation, respectively. The Durbin-Watson statistic value of the regression model is 2.03, which is very close to 2 indicating that there is no autocorrelation and the residuals in the model are independent. Therefore, the model satisfied the independence of residuals assumption. The third assumption is no multicollinearity, which means that there should not be strong correlations between predictor variables in the regression model. Since the regression model uses principal components that are uncorrelated, this inherently removes multicollinearity. However, I wanted to confirm this so I used the Variance Inflation Factor (VIF), which measures the degree of multicollinearity in the model. A VIF value of 1 implies there is no multicollinearity, whereas values above 5 or 10 imply significant multicollinearity. According to the VIF table, all principal components have a VIF value close to 1, indicating there is no multicollinearity in the model and the components are independent of each other. Therefore, the model satisfied the no multicollinearity assumption.

**G4. Provide the regression equation and discuss the coefficient estimates (REVISION FIXED)**

Regression Equation: *Y = Intercept + (Coefficient of X1) \* X1 + (Coefficient of X2) \* X2 + ... + (Coefficient of Xn) \* Xn*

Coefficients from our regression output:*Price = 306700 + 61890 \* PC1 - 24130 \* PC2 + 30420 \* PC3 + 4530 \* PC4 - 3077 \* PC7 - 3915 \* PC8 + 9282 \* PC10*

The intercept value of 306700 represents the starting predicted price when all principal components are set to zero. PC1 has a coefficient value of 61890, indicating that for each one-unit increase in PC1, the predicted housing price increases by $61,890. PC2 has a coefficient value of -24130, indicating that for each one-unit increase in PC2, the predicted housing price decreases by $24,130. PC3 has a coefficient value of 30420, indicating that for each one-unit increase in PC3, the predicted housing price increases by $30,420. PC4 has a coefficient value of 4530, indicating that for each one-unit increase in PC4, the predicted housing price increases by $4,530. PC7 has a coefficient value of -3070, indicating that for each one-unit increase in PC7, the predicted housing price decreases by $3,070. PC8 has a coefficient value of -3915, indicating that for each one-unit increase in PC8, the predicted housing price decreases by $3,915. PC10 has a coefficient value of 9282, indicating that for each one-unit increase in PC10, the predicted housing price increases by $9,282.

To summarize, the principal components with positive coefficient values are PC1, PC3, PC4, and PC10. These components increase predicted housing prices when their values increase. The principal components with negative coefficient values are PC2, PC7, and PC8. These components decrease predicted housing prices when their values increase. Since PC1 is the component that captures the most variance, organizations should focus on key variables that have high positive contributions in PC1 such as SquareFootage, SchoolRating, RenovationQuality, and PreviousSalePrice, as seen from the loadings matrix in part E1.

**G5. Discuss the model metrics**

The optimized regression model's R-squared value on the training set is 0.638, which means 63.8% of the variance in housing prices is explained by the model. This suggests that the principal components that were retained during the backward stepwise elimination process contribute meaningfully to explaining the variance in housing prices. The adjusted R-squared value on the training set is 0.637, which is only 0.001 lower than the R-squared value. This very small gap between R-squared and adjusted R-squared confirms that the retained principal components in the model are relevant and do not overly complicate the model because the adjusted R-squared value did not get penalized as much. Also, this small difference suggests that the model doesn't suffer significantly from overfitting.

The Mean Squared Error (MSE) on the training set is 8220717571.94, whereas the MSE on the test set is 7782198044.08. The training and test MSE values are highly similar, indicating that the optimized regression model performs equally well on both the training data and unseen test data. The close values also suggest that the model isn't overfitting to the training set, making it reliable for predicting house prices on new data.

**G6. Discuss the results and implications of your prediction analysis**

The optimized regression model shows strong generalization to new data and achieves reasonable prediction accuracy for housing prices, as indicated by similar training and test MSE values. It also helps that the seven principal components retained in the model through the backward stepwise elimination process significantly explain the variability in housing prices, as shown in the model's R-squared and adjusted R-squared values. These components have p-values below 0.05, making them statistically significant predictors of housing prices. The components with positive coefficient estimates are PC1, PC3, PC4, and PC10, meaning that they increase predicted housing prices when their values increase. The components with negative coefficient estimates are PC2, PC7, and PC8, meaning that they decrease predicted housing prices when their values increase. Regarding the model's assumptions, it has been verified that the regression model satisfied the assumptions of linearity, independence of residuals, and no multicollinearity.

Overall, the optimized regression model effectively predicts housing prices with reasonable accuracy and generalization, as indicated by its R-squared and adjusted R-squared values, similar MSE values for both training and test sets, and statistically significant coefficients. However, it is clear that the model can still be improved to achieve better predictive accuracy.

**G7. Recommend a course of action for the real-world organizational situation from part B1 based on your results and implications discussed in part E6**

A screenshot of a computer

Description automatically generated

I created a loading matrix for principal components (PC1, PC3, PC4 & PC10) that have positive coefficient estimates to focus on components that have a positive influence to housing prices. In PC1, the variables with high positive loadings are RenovationQuality, SchoolRating, SquareFootage, and PreviousSalePrice. In PC3, the variables with high positive loadings are CrimeRate and EmploymentRate. In PC4, the variables with high positive loadings are AgeOfHome, DistanceToCityCenter, and PropertyTaxRate. In PC10, the variables with high positive loadings are SquareFootage, AgeOfHome, and CrimeRate. These variables with high positive loadings should be prioritized when acting on the insights from PCA.

Based on the findings and insights from our regression model, a course of action that organizations like real estate companies or developers can perform is to focus on key features that drive housing prices to maximize their profitability. For example, since RenovationQuality has a significant positive relationship with housing prices, organizations should invest in high-quality renovations like upgrading interior and exterior features to increase property prices. Another example is prioritizing building larger properties because properties with larger square footage are valued more, as indicated by their positive loading. Organizations should also consider building properties in areas with low crime rates and high employment rates because these areas tend to be more desirable to potential buyers and command higher property values.

**H. Panopto Video**

Here’s the link to my Panopto Presentation: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=9488e1aa-2c9b-4491-9591-b23301655ca4>

**References**

Choueiry, G. (2019, October 26). *Understand Forward and Backward Stepwise Regression*. Quantifying Health. https://quantifyinghealth.com/stepwise-selection/