Boston Housing Report

Nature of the Selected Data for Processing:

This dataset contains information about various characteristics of **Boston's neighborhoods**. The **Boston dataset** includes **506 rows** and **14 columns**. Each row represents a **neighborhood (or town) in the Boston area**, and each column contains one of the following variables describing different characteristics of that area. Below is a description of each variable in the dataset:

1. **crim** – Crime rate per capita by town.
2. **zn** – Percentage of residential land zoned for lots larger than 25,000 square feet.
3. **indus** – Percentage of non-retail business acres per town.
4. **chas** – Dummy variable for the Charles River (1 if the neighborhood borders the river; 0 otherwise).
5. **nox** – Nitrogen oxide concentration (in parts per 10 million).
6. **rm** – Average number of rooms per dwelling.
7. **age** – Percentage of owner-occupied units built before 1940.
8. **dis** – Weighted distance to five major Boston employment centers.
9. **rad** – Index of accessibility to radial highways.
10. **tax** – Property tax rate per $10,000.
11. **ptratio** – Pupil-teacher ratio by town.
12. **black** – A calculation based on the proportion of the Black population in the town, using the formula:

    $$\mathbf{1000 \cdot (B - 0.63)^2}$$

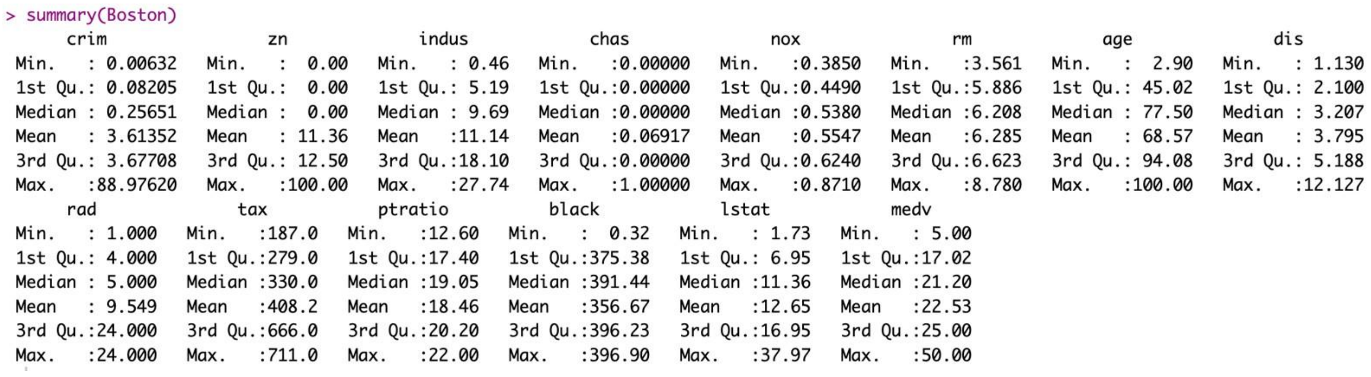
    where **B** represents the proportion of Black residents.
13. **lstat** – Percentage of lower socioeconomic status residents.
14. **medv** – Median value of owner-occupied homes (in thousands of dollars).

Solution:

The first step in working with the data was to examine the basic structure of the dataset. I used the summary(Boston) command in **RStudio**, which displayed the fundamental **statistical characteristics** of all variables in the **Boston dataset**.

> summary(Boston)

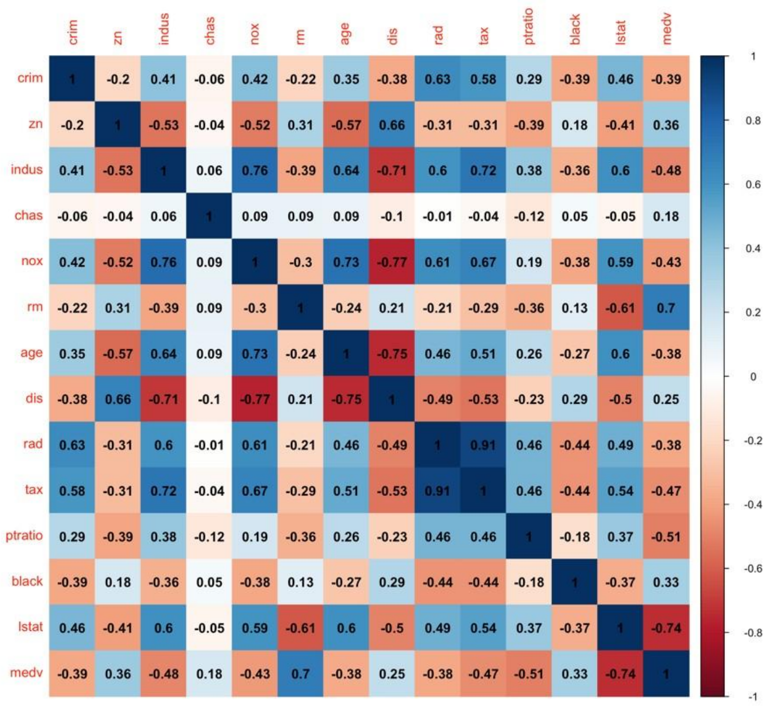
Based on this overview, I found that some variables, such as **crim** and **zn**, have a wide range of values. Additionally, the **chas** variable is binary. For this reason, I decided to **standardize the data** to eliminate differences in scale and improve the performance of future regression models.



```
> summary(Boston)
      crim                zn             indus            chas              nox               rm             age              dis        
 Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000   Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
 1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000   1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100  
 Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000   Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207  
 Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917   Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795  
 3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000   3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188  
 Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000   Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127  
      rad              tax           ptratio          black            lstat            medv      
 Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32   Min.   : 1.73   Min.   : 5.00  
 1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38   1st Qu.: 6.95   1st Qu.:17.02  
 Median : 5.000   Median :330.0   Median :19.05   Median :391.44   Median :11.36   Median :21.20  
 Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67   Mean   :12.65   Mean   :22.53  
 3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23   3rd Qu.:16.95   3rd Qu.:25.00  
 Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90   Max.   :37.97   Max.   :50.00  
```

After **standardizing the data** and creating the **correlation matrix**, we obtained insights into the correlation patterns between different variables in the **Boston dataset**. **Standardization** ensures that all variables have a **mean of 0** and a **standard deviation of 1**, meaning that correlations between variables are **not affected by differences in scale**.

Interpretation of the Correlation Matrix:



	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
crim	1	-0.2	0.41	-0.06	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39
zn	-0.2	1	-0.53	-0.04	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36
indus	0.41	-0.53	1	0.06	0.76	-0.39	0.64	-0.71	0.6	0.72	0.38	-0.36	0.6	-0.48
chas	-0.06	-0.04	0.06	1	0.09	0.09	0.09	-0.1	-0.01	-0.04	-0.12	0.05	-0.05	0.18
nox	0.42	-0.52	0.76	0.09	1	-0.3	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43
rm	-0.22	0.31	-0.39	0.09	-0.3	1	-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.7
age	0.35	-0.57	0.64	0.09	0.73	-0.24	1	-0.75	0.46	0.51	0.26	-0.27	0.6	-0.38
dis	-0.38	0.66	-0.71	-0.1	-0.77	0.21	-0.75	1	-0.49	-0.53	-0.23	0.29	-0.5	0.25
rad	0.63	-0.31	0.6	-0.01	0.61	-0.21	0.46	-0.49	1	0.91	0.46	-0.44	0.49	-0.38
tax	0.58	-0.31	0.72	-0.04	0.67	-0.29	0.51	-0.53	0.91	1	0.46	-0.44	0.54	-0.47
ptratio	0.29	-0.39	0.38	-0.12	0.19	-0.36	0.26	-0.23	0.46	0.46	1	-0.18	0.37	-0.51
black	-0.39	0.18	-0.36	0.05	-0.38	0.13	-0.27	0.29	-0.44	-0.44	-0.18	1	-0.37	0.33
lstat	0.46	-0.41	0.6	-0.05	0.59	-0.61	0.6	-0.5	0.49	0.54	0.37	-0.37	1	-0.74
medv	-0.39	0.36	-0.48	0.18	-0.43	0.7	-0.38	0.25	-0.38	-0.47	-0.51	0.33	-0.74	1

Figure 1. Correlation Matrix

Strong Negative Correlations:

- **dis** (distance to employment centers) & **indus** (industrial concentration) = -0.71
  → Areas closer to employment centers tend to have higher industrial activity, while suburban areas have less industrial development.
- **dis** & **nox** (pollution levels) = -0.77
  → Closer proximity to the city center is associated with higher air pollution, while suburban areas have cleaner air.
- **dis** & **age** (percentage of older buildings) = -0.75
  → Older buildings are more common in urban areas, whereas newer constructions are found farther from the city center.
- **medv** (median house price) & lstat (low socioeconomic status percentage) = -0.74
  → Areas with a higher percentage of lower-income residents tend to have lower median house prices.

This suggests that areas with **higher industrial concentration and pollution** are often **closer to employment centers**. Additionally, **older properties** tend to have **lower house prices** and are generally located **farther from employment hubs**.

Strong Positive Correlations:

- **tax** (property tax rate) **& rad** (highway accessibility index) = 0.91
  → Areas with better access to highways tend to have higher property tax rates, possibly due to increased infrastructure costs or higher demand for accessibility.
- **tax & indus** (industrial concentration) = 0.72
  → Neighborhoods with a higher percentage of industrial areas tend to have higher property tax rates, which may reflect zoning policies or the economic impact of industrial activity.
- **rm** (average number of rooms per dwelling) **& medv** (median house price) = 0.7
  → Larger houses (more rooms) are generally associated with higher median home prices, indicating that housing size is a strong factor in property valuation.

This means that areas with **higher property tax rates** tend to have **better access to radial highways** and a **higher concentration of industrial activity**. Additionally, **houses with a larger average number of rooms** are generally **associated with higher home prices**.

Relationship Between the Number of Rooms and House Prices:

Based on the pair plots, there is a clear upward linear trend between the median house price (medv) and the average number of rooms (rm). This indicates that houses with more rooms tend to be more expensive.

Relationship Between Socioeconomic Status (lstat) and House Prices:

The graphs show that areas with a higher percentage of lower-income residents (lstat) tend to have lower median house prices (medv). This suggests that neighborhoods with a lower socioeconomic status generally have more affordable housing.

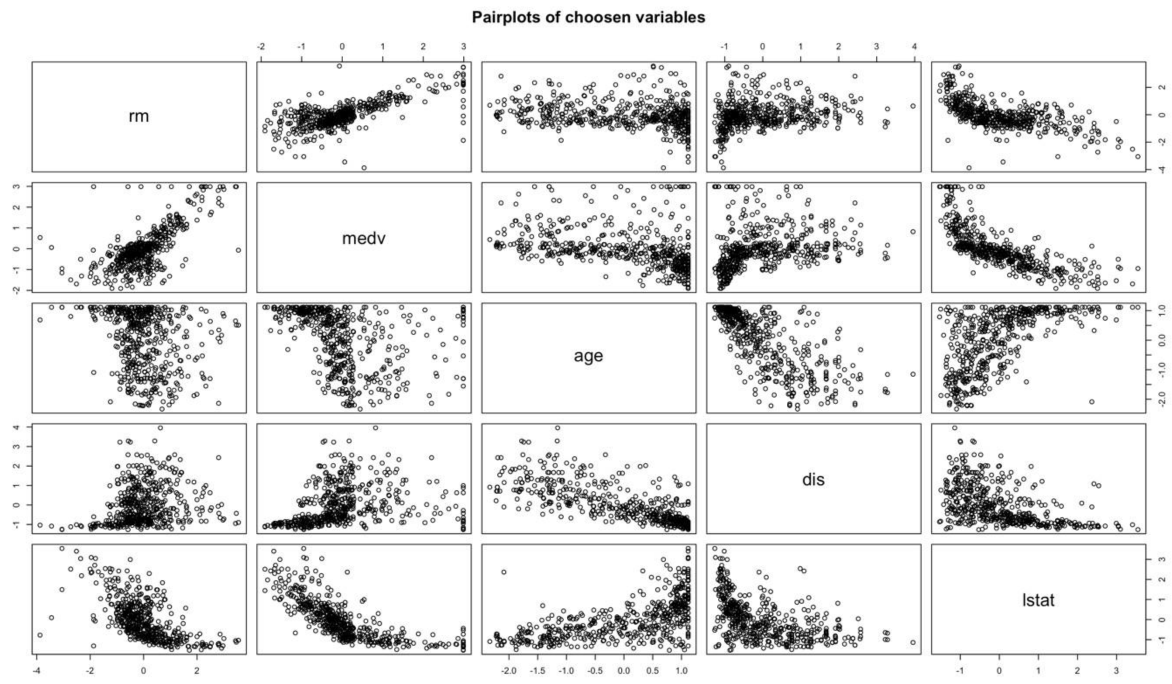


Figure 2. Pairplots

To better illustrate the relationships, a **3D scatterplot** was created, showing the dependence between:

- **medv** (median house price),
- **lstat** (percentage of lower-income residents),
- **rm** (average number of rooms per dwelling).

This visualization clearly highlights how **house prices (medv) increase with a higher number of rooms (rm)** while **decreasing in areas with a higher percentage of lower-income residents (lstat)**.

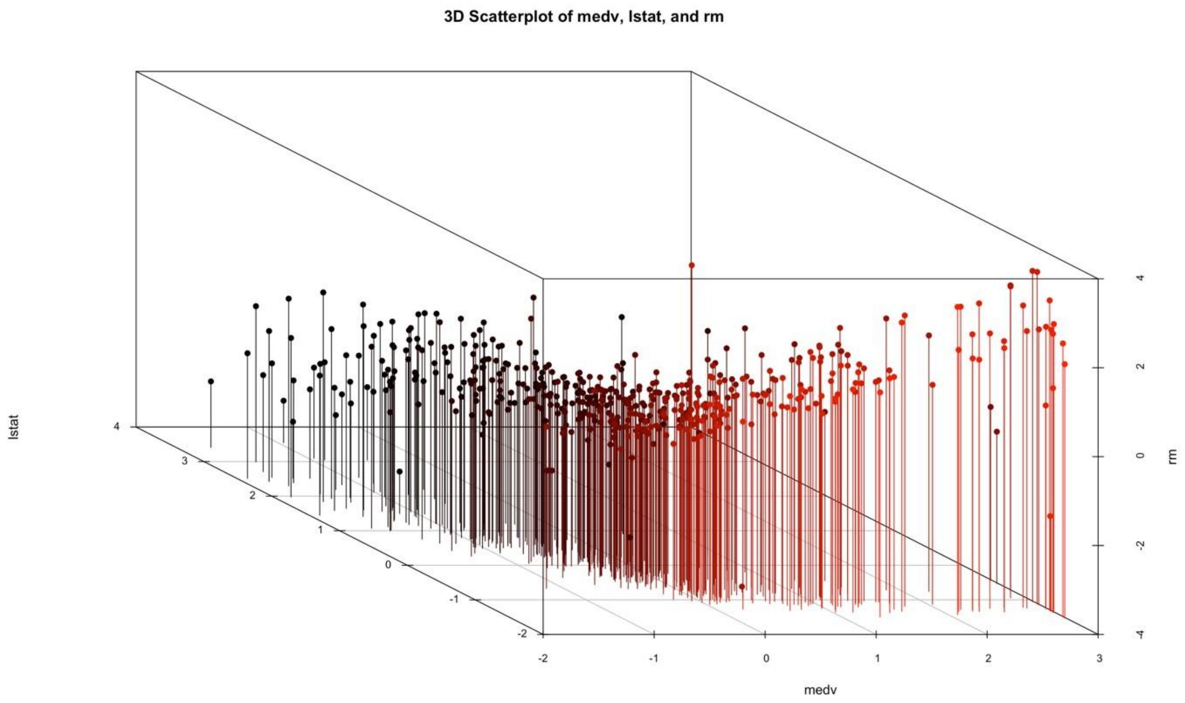


Figure 3. 3D-scatterplot mezi *medv*, *rm* a *lstat*

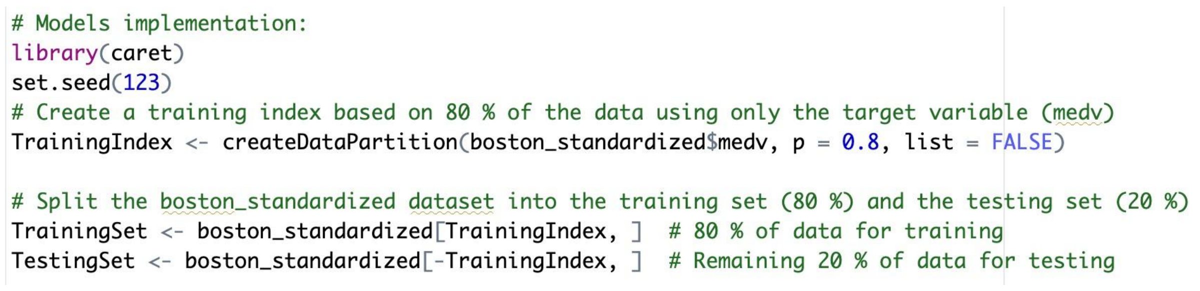
Regression models implementation:

Linear regression:

The first regression method chosen was **linear regression**. For implementation, the **previously standardized dataset** was used. The data was split into **two subsets** using the **'caret'** package:

- **80% of the data** was used for **training** (**TrainingSet**).
- The remaining **20%** was used for **testing** (**TestingSet**).

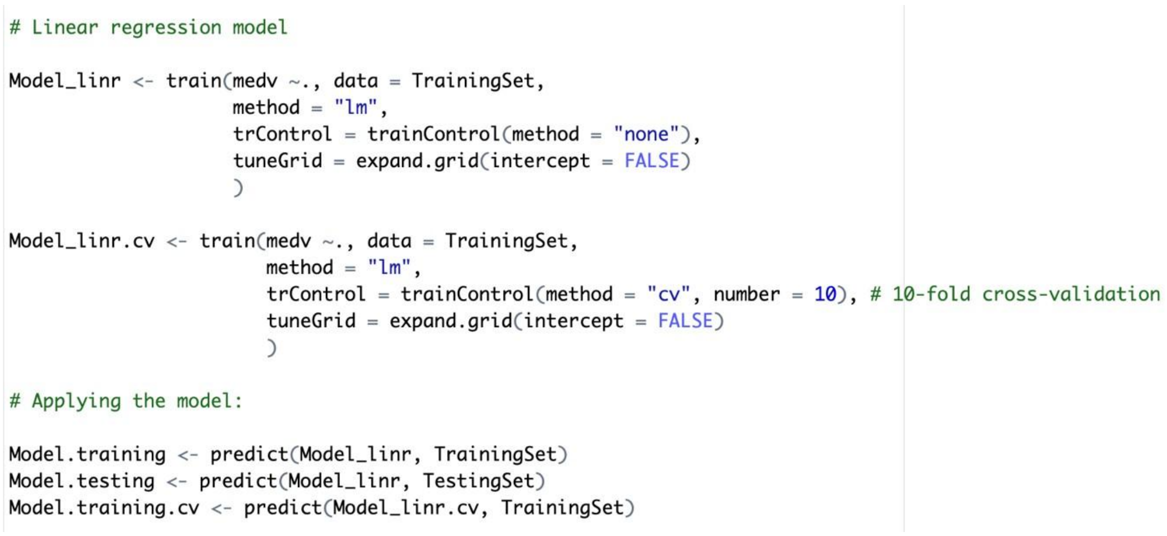
The **target variable** was **'medv'** (median house price), while all other variables were considered as **features** for prediction.



```
# Models implementation:
library(caret)
set.seed(123)
# Create a training index based on 80 % of the data using only the target variable (medv)
TrainingIndex <- createDataPartition(boston_standardized$medv, p = 0.8, list = FALSE)

# Split the boston_standardized dataset into the training set (80 %) and the testing set (20 %)
TrainingSet <- boston_standardized[TrainingIndex, ]  # 80 % of data for training
TestingSet <- boston_standardized[-TrainingIndex, ]  # Remaining 20 % of data for testing
```

After splitting the dataset, a **linear regression model** was created using functions from the **'caret'** package.

In addition to a **basic linear regression model**, a second model was implemented using **10-fold cross-validation**, which helps improve **model reliability** by evaluating its performance on multiple subsets of the data.



```
# Linear regression model

Model_linr <- train(medv ~., data = TrainingSet,
                    method = "lm",
                    trControl = trainControl(method = "none"),
                    tuneGrid = expand.grid(intercept = FALSE)
                    )

Model_linr.cv <- train(medv ~., data = TrainingSet,
                       method = "lm",
                       trControl = trainControl(method = "cv", number = 10), # 10-fold cross-validation
                       tuneGrid = expand.grid(intercept = FALSE)
                       )

# Applying the model:

Model.training <- predict(Model_linr, TrainingSet)
Model.testing <- predict(Model_linr, TestingSet)
Model.training.cv <- predict(Model_linr.cv, TrainingSet)
```

After applying the **linear regression model** to both the **training** and **test datasets**, the following results were obtained:

```
> print(RMSE_perfomance_train)            >print(R_square_train)
      [1] 0.512915                             [1] 0.7346007

> print(RMSE_perfomance_test) [1]          >print(R_square_test)
      0.4988922                                [1] 0.761151
```

The Root Mean Squared Error (RMSE) values were relatively low, approximately 0.51 for the training set and 0.50 for the test set. RMSE represents the average difference between predicted and actual values. Lower RMSE values indicate higher prediction accuracy.Since the RMSE values for both training and test sets are very similar, this suggests that the model generalizes well and **does not suffer from overfitting or underfittin**g.

The **R²** value for the training set is 0.7346, and for the test set, it is 0.7612. R² (coefficient of determination) indicates how well the model explains the variability in the data. Values around 0.73 – 0.76 suggest that the model explains approximately 73% to 76% of the variance in house prices. This is a solid result, meaning that the model can capture most of the key factors influencing housing prices while still allowing for some variability due to other influences.

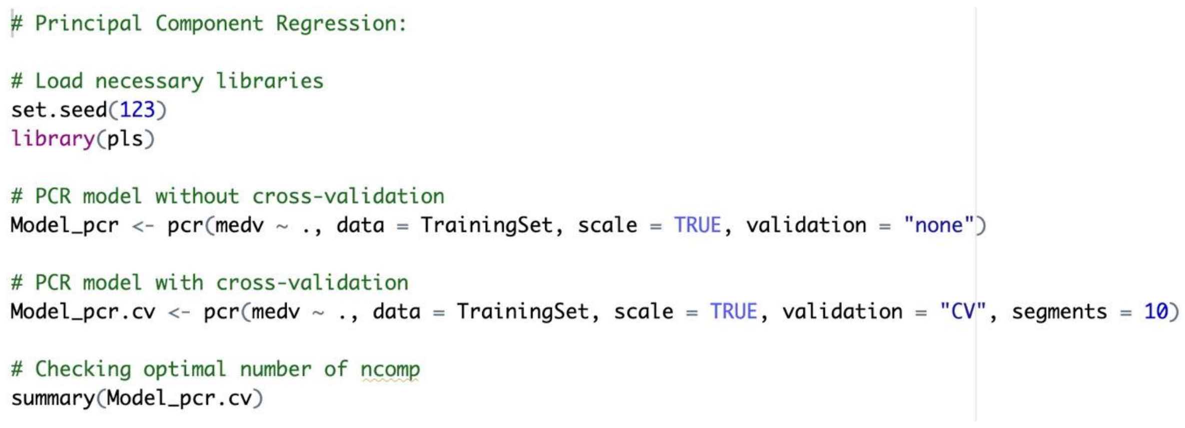
```
> print(RMSE_perfomance_train_cv)              > print(R_square_train_cv)
                [1] 0.5129162                               [1] 0.7346007
```

The model using cross-validation produced very similar results, confirming that the previous findings were not overly optimistic. Cross-validation helps verify that the model's performance is stable and not overly dependent on a specific data split.

Principal Component Regression (PCR):

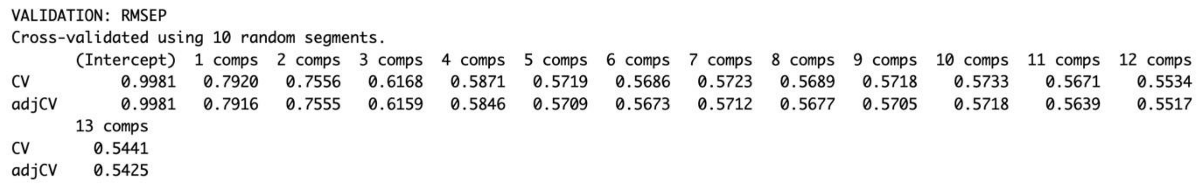
The next model I implemented was **Principal Component Regression (PCR)**, which combines **Principal Component Analysis (PCA)** with **linear regression**. I used the same standardized dataset, split into **training and test sets**, similarly to the linear regression model.

To implement the model, it was necessary to determine the **optimal number of principal components** to use. Therefore, after building the model, I examined detailed information using the following commands:



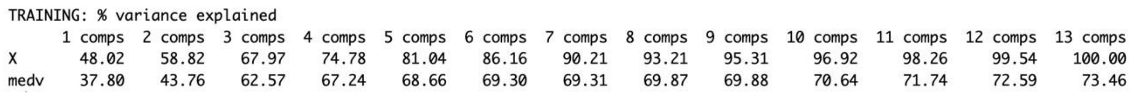
```r
# Principal Component Regression:

# Load necessary libraries
set.seed(123)
library(pls)

# PCR model without cross-validation
Model_pcr <- pcr(medv ~ ., data = TrainingSet, scale = TRUE, validation = "none")

# PCR model with cross-validation
Model_pcr.cv <- pcr(medv ~ ., data = TrainingSet, scale = TRUE, validation = "CV", segments = 10)

# Checking optimal number of ncomp
summary(Model_pcr.cv)
```

I found that the optimal number of components for this method would be **either 4 or 5**, as the cross-validation (CV) values remain relatively stable after reaching the fourth and fifth components. This suggests that adding more components does not significantly improve the model's performance.



```
VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
CV          0.9981   0.7920   0.7556   0.6168   0.5871   0.5719   0.5686   0.5723   0.5689   0.5718    0.5733    0.5671    0.5534
adjCV       0.9981   0.7916   0.7555   0.6159   0.5846   0.5709   0.5673   0.5712   0.5677   0.5705    0.5718    0.5639    0.5517
       13 comps
CV       0.5441
adjCV    0.5425
```

From the plot, it is also evident that after the **fifth component**, **81.04% of the data variability** is explained. This was the key reason for selecting **five components** as the optimal number for the model.



```
TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
X       48.02    58.82    67.97    74.78    81.04    86.16    90.21    93.21    95.31     96.92     98.26     99.54    100.00
medv    37.80    43.76    62.57    67.24    68.66    69.30    69.31    69.87    69.88     70.64     71.74     72.59     73.46
```

After completing the implementation of the model, I obtained the following results:

```
> print(RMSE_pcr_training) [1]          > print(RMSE_pcr_testing) [1]
      0.557398                                0.5247246

 > print(R2_pcr_training)                  > print(R2_pcr_testing)
     [1] 0.6865652                          [1] 0.7510312
```

The results indicate that the **PCR model** performs slightly worse on the training data compared to **classical linear regression**. For example, the **RMSE** for the training set is **higher in PCR (0.557 vs. 0.513)**, and the **R² value is lower (0.687 vs. 0.735)**. This suggests that the PCR model explains slightly less variability in the training data.

On the other hand, when examining the performance on the test set, the results for PCR and linear regression are **very similar**. The RMSE values are very alike (0.525 for PCR vs. 0.499 for linear regression), and the R² values are also close (0.751 for PCR vs. 0.761 for linear regression).

This suggests that while the **PCR model** has slightly lower adaptability to the **training data**, it performs well in **predicting new, unseen data**. To further validate this model, I also applied **cross-validation**, which yielded the following results:

```
> print(RMSE_pcr_training_cv)          > print(R2_pcr_training_cv)
     [1] 0.557398                          [1] 0.6865652
```

The cross-validation results are **consistent** with the original training set results without cross-validation. This indicates that the **PCR model** has **stable performance**, not only on a specific data split but also when tested repeatedly on different subsets of data.

In practice, **PCR is a useful tool when input variables are highly correlated and multicollinearity is a concern**. Although the model may not be as precise on the training data, its **robustness and ability to avoid overfitting** give it an advantage in **predicting new data**.

Conclusion:

Linear regression performs better on training data and can be a good choice when **multicollinearity is not a significant issue** in the dataset.

The **PCR model** is slightly less effective on training data but produces **very similar results on test data compared to linear regression**. This suggests that **PCR is a suitable approach when explanatory variables are highly correlated, as it helps reduce overfitting**.

Depending on the nature of the data, **linear regression** is recommended if **multicollinearity is not a concern**, while **PCR is preferable in cases where multicollinearity is an issue**, as it provides **robust predictions for new data**.

Source

Harrison, D. and Rubinfeld, D.L. (1978) Hedonic prices and the demand for clean air. *J. Environ. Economics and Management* **5**, 81–102.

Belsley D.A., Kuh, E. and Welsch, R.E. (1980) *Regression Diagnostics. Identifying Influential Data and Sources of Collinearity.* New York: Wiley.