Hands-on Assignment 1

Team 2: Tom Bailey, Patricia Basallo Fraga, Elie Bassil, Isabel Hernandez-Marquez, Lena Kennedy and Finn Lepree

**Discuss the purposes/objectives of the Boston housing data mining problem and the specific data mining tasks that are relevant for this module**

The Boston Housing data mining problem goal is to predict the median value of owner-occupied homes MEDV in many Boston neighborhoods, based on socioeconomic and environmental variables. We have two goals in this problem. First, we need to develop accurate predictive models, and second gain a better understanding of the relationships between housing prices and multiple influencing factors.

The main goal is to create a model that accurately predicts the median home value, based on a variety of predictor variables like crime rates, the number of rooms, accessibility to highways, etc. We can make informed decisions about property investments or urban development policies by understanding these relationships.

Our second goal is to investigate the impact of socio-economic factors like LSTAT- percentage of lower status population). We also have environmental factors like (NOX-pollution levels, or CHAS- proximity to charles river) on home prices. Analysis like this provides insights into what drives home values in Boston.

Another goal is to improve model accuracy and interpretability. Primarily selecting the most relevant features, transforming variables when needed, and making sure that the results are not only accurate but also easy to interpret by stakeholders.

The purpose of data exploration is to understand the structure and distribution of the dataset and identify key variables that are influencing the MEDV. Its use is to generate summary statistics for all variables, and to use visualizations to explore relationships between variables like RM and MEDV, additionally to detect outliers and/or anomalies. Data cleaning and preprocessing help us ensure that the dataset is ready for analysis by handling missing or inconsistent data. It is also instrumental in preparing variables for modeling. It is tasked with identifying and handling missing values or outliers. Perform transformations, such as normalizing skewed variables. Dimension reduction is to select the most important features and reduce repetitions in the dataset, leading to an improvement in model performance. The purpose of model building is to develop predictive models to estimate MEDV based on selected features. We used it to experiment with different types of models and compare their performances.

**Describe the dataset in detail**

This dataset has 506 total rows of observations (the 1st row contains the variable names). In this dataset, we can find 14 columns or variables that provide different characteristics for each record and a more robust perspective. This dataset contains predictor variables that can help us predict the median value of homes in different Boston neighborhoods. You may find the following variable names in the dataset: CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD, TAX, PTRATIO, LSTAT, MEDV, and CAT.MEDV.

Out of these 14 variables, only 1 variable or CAT.MEDV was created to categorize median value into two categories: high and low. Below you will find details or a description of each variable and its contents.

|  |  |
| --- | --- |
| **Variable** | **Variable Description and Detail** |
| CRIM | A (numerical) per capita crime rate value by town. |
| ZN | A (numerical) percentage or proportion of residential land zoned for lots over 25,000 ft2. |
| INDUS | A (numerical) percentage or proportion of land occupied by non-retail business acres per town. |
| CHAS | A (categorical) variable that indicates whether or not the tract is near the Charles River. The value of “1” means that the tract bounds the river while the value of “0” means otherwise. |
| NOX | A (numerical) variable that specifies the nitric oxide concentration (in parts per 10 million). |
| RM | A (numerical) variable indicating the average number of rooms in a dwelling. |
| AGE | A (numerical) percentage or proportion of owner-occupied units built prior to 1940. |
| DIS | A (numerical) variable of the weighted distances to five Boston employment centers. |
| RAD | A (numerical) variable of the index of accessibility to radial or major highways. |
| TAX | A (numerical) variable of the full-value property tax rate per $10,000.  Sequential / Dependency Note: It is important to figure out the property value first because it is fundamental in figuring out the TAX value. |
| PTRATIO | A (numerical) variable of the pupil-to-teacher ratio by town. |
| LSTAT | A (numerical) percentage or proportion of lower status of the population. |
| MEDV | A (numerical and target) variable of median value of owner-occupied homes in $1000s. |
| CAT.MEDV | An additional (categorical) variable to the original dataset, identifying observations that have a median value of owner-occupied homes in tract above $30,000 with a value of “1” or with a value “0” if otherwise. |

**Discuss what you have done in terms of data exploration, data cleaning, and preprocessing. What you have done in terms of variable selection (dimension reduction), variable transformation, etc. Please provide details such as screenshots of ASDM or R procedures, graphs, outputs, justifications of your actions, etc**

**Data Exploration**  
We started by exploring the Boston housing dataset to understand the structure and relationships among the variables, with a big focus on how LSTAT (percentage of lower-status population) and RM (median value of owner-occupied homes). We initiate our exploration by creating several plots.

The first plot is a scatter plot named “g1” that helps reveal any patterns between the numerical variables: LSTAT (on the x-axis) and MEDV (on the y-axis). Navy colored dots and an alpha parameter of 0.5 helps the user visually see the dots better with a contrasting color and a semi-transparency view to help identify dots that may be clustering or overlapping. This plot visually demonstrates the potential, positive or negative, relationship between the percentage of lower-status populations (LSTAT) and the median home values (MEDV).   
  
**Figure 1**. Scatter plot of MEDV and LSTAT  
A graph with blue dots

Description automatically generated

The second plot is a bar chart named “g2” that helps show a single statistic (average) across groups. In our bar chart, we can see that the mean of MEDV (numerical variable we created as MeanMEDV) is on the y-axis and CHAS is on the x-axis (categorical variable). Before creating this chart, we had to find the mean of the MEDV (median home values) for each category of the CHAS variable (“1” means that the tract bounds the river while the value of “0” means otherwise) with an aggregate function by grouping the CHAS variable, creating a new data frame (MEDV.per.CHAS) to hold the mean values, renaming columns, and using factor (for categorical variables in R) to help treat this variable as a categorical variable. For the aesthetics of the plot, we added the fill color to be based on the CHAS variable and distinguish which homes are near the river (teal) or not (pink). We make sure to set the height of the bars to correspond with the MeanMEDV values by setting the stat = “identity” (of the y-values or MeanMEDV) instead of the default (count). This plot helps us compare how proximity to the Charles River (CHAS) may affect the average home prices (MeanMEDV).   
  
**Figure 2**. Bar Chart of MeanMEDV and CHAS  
  
A screenshot of a graph

Description automatically generated

The third plot is a histogram named “g3” that helps show the distribution of MEDV (median home prices) and any potential skewness or outliers in the dataset. We can view the frequencies of all x values with a series of connected bars and see how there is a skewed distribution coming through the 9 bins. We can also visually see how homes may cluster around a (low or high) price range or if values are spread across a wide range of prices. This plot can help us compare groups that may be identifiable from the distribution of data and potential transformations for analysis.

**Figure 3.** Histogram of MEDV **A graph of a number of values

Description automatically generated**

The fourth plot is a heatmap named “g4” that helps us show pairwise correlations between 14 variables in a two-dimensional table with a red-blue color palette. Each tile in the heatmap is a correlation between two variables and the color indicates the type of correlation (positive or negative) and strength (high or low). For example, a darker blue represents negative correlations and a darker red represents positive correlations. We decided to round correlation values to 2 decimals because it may be easier to interpret and read. We made sure to transform our matrix into the right format with the melt() function, color intensity corresponds to the correlation value, and implement a red-blue color palette. This plot helps us identify variables worth exploring and which highly correlated variables may be redundant.

**Figure 4.** Correlation Matrix of Numeric Variables

A screenshot of a graph

Description automatically generated  
  
  
**Data Cleaning**

We assessed the Boston housing dataset for any missing values by using the is.na() function and sum() function to count how many values are missing. We checked for missing values in the LSTAT, RM, and MEDV columns. Our script returned a “0” for these variables which means we didn’t have any missing values, our data is relatively clean, and doesn’t require any handling for missing data.

We did not use Principal Component Analysis in this case, as we did not consider the data needed for this kind of analysis. There were only two features and there was no need for dimension reduction.

**Model Training and Evaluation Process**

*Model 1*  
We create a test and training set to predict housing prices based on two variables: LSTAT and RM. We set a seed to have random processes (for splitting data) and avoid inconsistent results. Next, we randomly select 60% of rows (from LSTAT, RM, and MEDV) we intend to use for the training set. A test (holdout) set is created to test the accuracy of our model with rows not included in the training set (remaining 40%). Printing the count of rows in each set helps us validate how we split our data.

Training the model is done by using a lm() function with LSTAT and RM variables as predictors while MEDV is the target variable. This model tries to predict MEDV based on LSTAT and RM using the training data (*train.df*). We make sure to store the trained regression model in *reg*.  
  
Evaluating the model on the training set by comparing the actual and predicted values is important. We use the data frame, *train.res*, to store the actual MEDV values, predicted values, and the residuals (differences between actual and predicted values) for the training set to evaluate the model’s performance on the training data.

Making predictions on the holdout set is done with the predict() function. The trained regression model named *reg* is used to predict MEDV for the test set based on the LSTAT and RM values in the holdout set. The predicted MEDV values for the holdout data are stored in *pred.* We then format the holdout set results to measure and focus on potential prediction errors by looking at the absolute error relative to the actual value (|actual - predicted | / actual). Visualizing the distribution of prediction errors from the holdout (test) set is done with the help of a boxplot. The boxplot demonstrates how the errors (and assess any outliers) are spread in the test set and how the model’s predictions deviate from the actual values.

Lastly, we calculate some accuracy measures like root-mean-squared error (RMSE), mean error (ME), and mean absolute error (MAE) to evaluate the performance of the prediction model on both training and test sets. With these measures, we can assess differences in how predicted values deviate from the actual values and potential overfitting.   
  
*Model 2*  
We create a new yet familiar linear regression model using only the LSTAT variable to predict the MEDV. In this model, RM is no longer included. Relatively, the focus is now on understanding the potential impact may be when using fewer predictor variables on model performance.

The model is trained on the training data, *train.df*, using the lm() function. However, we make sure to not include the RM variable in the training data set. After training, we stored the fitted values in *train.res* which also contains actual MEDV, and residuals.

To make predictions, we use the test data set, *holdout.df*, with the predict() function. Similarly to model 1, we store predictions in *pred* and the results are formatted into the *holdout.res* data frame. Calculating (and printing) the prediction error for model 2 helps us understand the variability in prediction accuracy. Next, we create a box chart plot to visualize the distribution of errors. In this case, there is a slight increase in mean error when comparing with model 1.

Lastly, we compute the same accuracy measures for model 2: Mean Error (ME), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). In comparison to model 1, removing the RM variable seems to reduce the model’s predictive power because there is an overall increase in error (when comparing ME, RMSE, and MAE).

**Discuss what you have learned about the Boston housing data from the data exploration and dimension reduction procedures that you have performed**

Many important insights about the Boston housing dataset have emerged, through the data exploration and dimension reduction. Two variables emerged as significant predictors for the target variable, MEDV. The negative correlation between LSTAT and MEDV indicates that neighborhoods with a higher percentage of lower-status individuals tend to have lower home values. We visualized this relationship through a scatter plot where LSTAT showed a clear downward trend with respect to MEDV.

The positive correlation between RM and MEDV was confirmed by both correlation matrix and scatter plot. This is a crucial predictor in the model due to more rooms in a home being generally associated with higher home values. The histogram of MEDV showed that most homes are clustered around values between $20000 and $30000, with fewer homes at higher prices. This skewed distribution is important because the skewness can affect the performance of regression models, with a focus on linear models that assume normally distributed residuals. Through the heatmap of correlations, it was evident that certain variables are highly correlated with each other. One example, TAX (property tax rate) and RAD (index of accessibility to radial highways) had a high positive correlation of 0.91. This suggests that these variables might not both be needed in a model and using them has the potential to cause multicollinearity issues. This is where predictors are correlated with each other rather than with the target variable.

We have variables like CRIM (crime rate) and TAX that had extreme values, that we retained during the data exploration phase. By retaining outliers we can allow the model to account for real-world extremes, but at the same time, its presence might affect model performance by increasing the variance in predictions. By selecting only two predictors, LSTAT and R, for the initial regression model, we were able to create a simplified model that explained much of the variation in MEDV.

This reduction improved our model’s interpretability and made it easier to assess the impact of each variable on home values. In the first model where we included both LSTAT and RM, the performance metrics indicated reasonably good accuracy. The RMSE on the test set was 5.63 and the MAE was 3.90. This would suggest that the model predicted MEDV with a moderate level of accuracy. In the second model, which only used LSTAT as the predictor, the performance was slightly worse. The RMSE was 6.24 and the MAE was 4.48, this indicates a larger average error in the predictions compared to the model that included RM. The increase in error confirms that RM is crucial in explaining home values. The model's predictive power decreases significantly without it. We evaluated the percent error of predictions across models, with the mean percent error being around 19.7% when both LSTAT and RM were included. The percent error increased slightly to 20.54% when we only used LSTAT. This increase shows the value of adding RM in the model. We selected both LSTAT and RM because it was driven by strong correlations with MEDV and their ability to explain much of the variation in home prices.

**R-Script**

# Install devtools package

# install.packages("devtools")

# Load Libraries

library(devtools)

library(ggplot2)

library(gridExtra)

library(mlba)

library(reshape2)

#install.packages("gridExtra")

#get housing dataset

#install\_github("gedeck/mlba/mlba", force=TRUE)

#install package for getting housing dataset

housing.df <- mlba::BostonHousing

colnames(housing.df)

#plot percentage of lower status of the population (X) and Median value of owner occupied homes

g1 <- ggplot(housing.df) + geom\_point(aes(x=LSTAT, y=MEDV), colour="navy", alpha=0.5)

# compute meanMEDV per CHAS0

MEDV.per.CHAS <-aggregate(housing.df$MEDV, by=list(housing.df$CHAS), FUN=mean)

names(MEDV.per.CHAS) <- c("CHAS", "MeanMEDV")

MEDV.per.CHAS$CHAS <- factor(MEDV.per.CHAS$CHAS)

# plot Median value by near the Charles river or not

g2 <- ggplot(MEDV.per.CHAS) + geom\_bar(aes(x=CHAS, y=MeanMEDV, fill=CHAS), stat="identity")

# distribution of median home values

g3 <- ggplot(housing.df) + geom\_histogram(aes(x=MEDV), bins=9) + ylab("Count") +

  ggtitle("Distribution of Median Home Values (MEDV)")

# simple heatmap of correlations

cor.mat <- round(cor(housing.df),2)

melted.cor.mat <- melt(cor.mat)

g4 <- ggplot(melted.cor.mat, aes(x=Var1, y=Var2, fill=value)) + geom\_tile() + xlab("") +ylab("") + scale\_fill\_distiller(palette="RdBu", limits=c(-1, 1))

# create grid of visualizations

grid.arrange(g1, g2, g3, g4)

A screenshot of a graph

Description automatically generated

# lets count null values are in each of the columns we intend to use to predict

print(sum(is.na(housing.df$LSTAT))) # Returns 0

print(sum(is.na(housing.df$RM))) # Returns 0

print(sum(is.na(housing.df$MEDV))) # Returns 0

# this means the data is relatively clean

#we will now create a test and training set

set.seed(1) # keep same partitions when re running

#select rows we intend to use

train.rows <- sample(rownames(housing.df), nrow(housing.df)\*.6)

#select data from rows we intend to use

train.df <- housing.df[train.rows, c( "LSTAT", "RM", "MEDV")]

# create holdout rows or test set, this is the data we will use to test the accuracy of our model

holdout.rows <- setdiff(rownames(housing.df), train.rows)

holdout.df <- housing.df[holdout.rows, c( "LSTAT", "RM", "MEDV")]

#print counts

print(nrow(holdout.df)) #203

print(nrow(train.df)) #303

#train the model

reg <- lm(MEDV ~ ., data=train.df[, c( "LSTAT", "RM", "MEDV")])

# format data in a way that is easier to test accuracy

train.res <- data.frame(actual=train.df$MEDV, predicted= reg$fitted.values, residuals=reg$residuals )

#predict

pred <- predict(reg, newdata=holdout.df[, c( "LSTAT", "RM", "MEDV")])

# format data in a way that is easier to test accuracy

holdout.res <- data.frame(actual=holdout.df$MEDV, predicted=pred, residuals=holdout.df$MEDV-pred, percent\_error=abs((holdout.df$MEDV - pred)/holdout.df$MEDV))

boxplot(holdout.res$percent\_error,

        main = "Boxplot of Percent Errors",

        ylab = "Percent Error (%)")

print(mean(holdout.res$percent\_error)\*100) #19.7%

A graph of percent error

Description automatically generated

library(caret)

#compute metrics on training set

data.frame(ME = round(mean(train.res$residuals), 5),

           RMSE = RMSE(pred=train.res$predicted, obs=train.res$actual),

           MAE = MAE(pred=train.res$predicted,obs=train.res$actual))

#training set

# ME     RMSE      MAE

#   0      5.459931  3.951087

data.frame(ME = round(mean(holdout.res$residuals), 5),

           RMSE = RMSE(pred=holdout.res$predicted, obs=holdout.res$actual),

           MAE = MAE(pred=holdout.res$predicted,obs=holdout.res$actual))

#holdout set

# ME            RMSE      MAE

#  0.13856   5.627824  3.902462

###########################################################################

#train the model (this time with no rooms)

reg <- lm(MEDV ~ ., data=train.df[, c( "LSTAT", "MEDV")])

# format data in a way that is easier to test accuracy

train.res <- data.frame(actual=train.df$MEDV, predicted= reg$fitted.values, residuals=reg$residuals )

#predict

pred <- predict(reg, newdata=holdout.df[, c( "LSTAT", "MEDV")])

# format data in a way that is easier to test accuracy

holdout.res <- data.frame(actual=holdout.df$MEDV, predicted=pred, residuals=holdout.df$MEDV-pred, percent\_error=abs((holdout.df$MEDV - pred)/holdout.df$MEDV))

boxplot(holdout.res$percent\_error,

        main = "Boxplot of Percent Errors (LSTAT ONLY)",

        ylab = "Percent Error (%)")

print(mean(holdout.res$percent\_error)\*100) #20.54186

A graph with a bar and a line

Description automatically generated with medium confidence

library(caret)

#compute metrics on training set

data.frame(ME = round(mean(train.res$residuals), 5),

           RMSE = RMSE(pred=train.res$predicted, obs=train.res$actual),

           MAE = MAE(pred=train.res$predicted,obs=train.res$actual))

#training set

# ME  RMSE      MAE

# 0   6.184536 4.496082

data.frame(ME = round(mean(holdout.res$residuals), 5),

           RMSE = RMSE(pred=holdout.res$predicted, obs=holdout.res$actual),

           MAE = MAE(pred=holdout.res$predicted,obs=holdout.res$actual))

#holdout set

# ME      RMSE      MAE

# 0.33432 6.241676  4.480597