Data Mining

Case Study 1

1a. Create a boston_df data frame by uploading the original data set into Python. Determine and present in this report the data frame dimensions, i.e., number of rows and columns from Python, and briefly explain these numbers.

The data set BostonHousing.csv has 506 rows and 14 columns.

```
# Create data frame from the original data set.
boston_df = pd.read_csv('BostonHousing.csv')

# Determine dimensions of dataframe.
boston_df.shape # It has 506 rows and 14 columns.
```

```
(506, 14)
```

1b. Display in Python the column titles and present them in your report. If some of them contain two (or more) words, convert them into one-word titles, and present the modified titles in your report.

The Columns of the report are 'CRIME', 'ZONE', 'INDUST', 'CHAR RIV', 'NIT OXIDE', 'ROOMS', 'AGE', 'DISTANCE', 'RADIAL', 'TAX', 'ST RATIO', 'LOW STAT', 'MVALUE', 'C MVALUE'

```
# Display the column names.
boston_df.columns
```

```
Index(['CRIME', 'ZONE', 'INDUST', 'CHAR RIV', 'NIT OXIDE', 'ROOMS', 'AGE',
       'DISTANCE', 'RADIAL', 'TAX', 'ST RATIO', 'LOW STAT', 'MVALUE',
       'C MVALUE'],
      dtype='object')
```

After converting columns that have two or more words you get the below result.
Converted Columns:
'CRIME', 'ZONE', 'INDUST', 'CHAR_RIV', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE', 'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE', 'C_MVALUE'.

```
In [4]: print('Modified column titles with no space and one word for titles:')
        boston_df.columns = [s.strip().replace(' ', '_') for s in boston_df.columns]
        boston_df.columns

        Modified column titles with no space and one word for titles:

Out[4]: Index(['CRIME', 'ZONE', 'INDUST', 'CHAR_RIV', 'NIT_OXIDE', 'ROOMS', 'AGE',
               'DISTANCE', 'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE',
               'C_MVALUE'],
              dtype='object')
```

1c. Display in Python column data types and present them in your report. If some of them are listed as "object', briefly explain that in your report, convert them into dummy variables, and provide in your report the modified list of column titles with dummy variables.

There are two variables listed as object those are "CHAR_RIV" and "C_MVALUE".

```
boston_df.dtypes

CRIME         float64
ZONE          float64
INDUST        float64
CHAR_RIV       object
NIT_OXIDE     float64
ROOMS         float64
AGE           float64
DISTANCE      float64
RADIAL          int64
TAX             int64
ST_RATIO      float64
LOW_STAT      float64
MVALUE        float64
C_MVALUE       object
dtype: object
```

"CHAR_RIV" has categories 'N' and 'Y'.

```
ı [7]:
boston_df.CHAR_RIV  = boston_df.CHAR_RIV.astype('category')

# Display category classes and category type.
print(' ')
print('Category levels and changed variable type:')
print(boston_df.CHAR_RIV.cat.categories)
print(boston_df.CHAR_RIV.dtype)

Category levels and changed variable type:
Index(['N', 'Y'], dtype='object')
category
```

“C_MVALUE” has categories 'No' and 'Yes'.

```
In [8]:
boston_df.C_MVALUE  = boston_df.C_MVALUE.astype('category')

# Display category classes and category type.
print(' ')
print('Category levels and changed variable type:')
print(boston_df.C_MVALUE.cat.categories)
print(boston_df.C_MVALUE.dtype)
```

```
Category levels and changed variable type:
Index(['No', 'Yes'], dtype='object')
category
```

Converting “CHAR_RIV” and “C_MVALUE” into dummy variables.

```
boston_df = pd.get_dummies(boston_df, prefix_sep='_',
                           drop_first=True)
boston_df.columns
```

```
Index(['CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE',
       'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE', 'CHAR_RIV_Y',
       'C_MVALUE_Yes'],
      dtype='object')
```

The modified list of column titles with dummy variable is ['CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE', 'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE', 'CHAR_RIV_Y', 'C_MVALUE_Yes]

Data Types after conversion:

```
boston_df.dtypes
```

```
CRIME           float64
ZONE            float64
INDUST          float64
NIT_OXIDE       float64
ROOMS           float64
AGE             float64
DISTANCE        float64
RADIAL            int64
TAX               int64
ST_RATIO        float64
LOW_STAT        float64
MVALUE          float64
CHAR_RIV_Y        uint8
C_MVALUE_Yes      uint8
dtype: object
```

1d. Display in Python the descriptive statistics for all columns in the modified boston_df data frame (after converting to one-word titles and dummy variables). Check if there are missing records (values) in the columns. Present the table with descriptive statistics in 2 your report, and comment about the missing values. You don't need to comment on the values of outliers (min/max) or their extreme values.

Descriptive statistics for all columns:

	CRIME	ZONE	INDUST	NIT_OXIDE	ROOMS	AGE	DISTANCE	RADIAL	TAX	ST_RATIO	LOW_STAT	MVALUE	CHAR_RIV_Y	C_MVALUE_Yes
count	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00	506.00
mean	3.61	11.36	11.14	0.55	6.28	68.57	3.80	9.55	408.24	18.46	12.65	22.53	0.07	0.17
std	8.60	23.32	6.86	0.12	0.70	28.15	2.11	8.71	168.54	2.16	7.14	9.20	0.25	0.37
min	0.01	0.00	0.46	0.38	3.56	2.90	1.13	1.00	187.00	12.60	1.73	5.00	0.00	0.00
25%	0.08	0.00	5.19	0.45	5.89	45.02	2.10	4.00	279.00	17.40	6.95	17.02	0.00	0.00
50%	0.26	0.00	9.69	0.54	6.21	77.50	3.21	5.00	330.00	19.05	11.36	21.20	0.00	0.00
75%	3.68	12.50	18.10	0.62	6.62	94.07	5.19	24.00	666.00	20.20	16.96	25.00	0.00	0.00
max	88.98	100.00	27.74	0.87	8.78	100.00	12.13	24.00	711.00	22.00	37.97	50.00	1.00	1.00

There are No missing Values in the columns as the count is the same for all columns.
Alternative way to check for missing values is using isnull():

```
boston_df.isnull().sum()
```

```
CRIME           0
ZONE            0
INDUST          0
NIT_OXIDE       0
ROOMS           0
AGE             0
DISTANCE        0
RADIAL          0
TAX             0
ST_RATIO        0
LOW_STAT        0
MVALUE          0
CHAR_RIV_Y      0
C_MVALUE_Yes    0
dtype: int64
```

2a. Develop in Python and present in your report outcome and predictor variables, partition the data set (80% for the training partition, and 20% for the validation partition), and train the multiple linear regression model using LinearRegression() with the training data set. Identify and display in Python intercept and regression coefficients of this model. Provide these coefficients in your report and present the mathematical equation of this linear regression model.

The outcome variable is “MVALUE” which is a numeric variable.

The predictor variables are 'CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE', 'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'CHAR_RIV_Y', 'C_MVALUE_Yes'.

```
# Identify predictors and outcome of the regression model.
predictors = ['CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE',
       'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'CHAR_RIV_Y',
       'C_MVALUE_Yes']
outcome = 'MVALUE'
```

In order to avoid overfitting we split the data into a training partition and a validation partition. The proportion of the training partition is 80% whereas, the proportion of the validation partition is 20%.

```
# Identify X and y variables for regression and partition data
# using 80% of records for training and 20% for validation
# (test_size=0.2).
X = boston_df[predictors]
y = boston_df[outcome]
train_X, valid_X, train_y, valid_y = train_test_split(X, y, test_size=0.2, random_state=1)
```

Parameters:

Intercept: 46.41

Below are the coefficients for each predictor:

```
Regression Model for Boston Housing Training Set

Intercept:  46.41
       Predictor  Coefficient
0          CRIME        -0.13
1           ZONE        -0.01
2         INDUST         0.11
3      NIT_OXIDE       -17.12
4          ROOMS         0.64
5            AGE        -0.01
6       DISTANCE        -0.70
7         RADIAL         0.19
8            TAX        -0.01
9       ST_RATIO        -0.60
10      LOW_STAT        -0.47
11    CHAR_RIV_Y         2.17
12  C_MVALUE_Yes        11.66
```

Regression Model Equation:

MVALUE = 46.41 -0.13 CRIME -0.01 Zone + 0.11 INDUST -17.12 NIT_OXIDE + 0.64 ROOMS -0.01 AGE -0.70 DISTANCE + 0.19 RADIAL -0.01 TAX -0.60 ST_RATIO -0.47 LOW_STAT + 2.17 CHAR_RIV_Y + 11.66 C_MVALUE_Yes

2b. Using the multiple regression model, identify in Python predictions for validation and training predictors (valid_X and train_X). Based on these predictions, identify and display in Python R 2 and adjusted R2 performance measures for training and validation partitions. Present and compare these performance measures in your report and explain if there is a possibility of overfitting.

```
Prediction Performance Measures for Training Set
r2 :  0.836
Adjusted r2 :  0.831
AIC :  2220.32
BIC :  2280.34

Prediction Performance Measures for Validation Set
r2 :  0.851
adjusted r2 :  0.829
AIC :  593.83
BIC :  633.2
```

<u>Prediction performance measures for training set:</u>

R2 is 0.836

Adjusted r2 is 0.831

<u>Prediction performance measures for validation set:</u>

R2 is 0.851

Adjusted R2 is 0.829

These results indicate that there is no overfitting, as both partitions show similar and high R2 and adjusted R2 values.

Additionally, the validation set outperforms the training set in terms of performance measures, suggesting that there is no over-prediction or over-fitting in the model.

2c. Identify and display in Python the common accuracy measures for training and validation data set (predictions). Provide and compare these accuracy measures in your report and explain again a possibility of overfitting.

```
Accuracy Measures for Training Set - All Variables

Regression statistics

                      Mean Error (ME) : -0.0000
       Root Mean Squared Error (RMSE) : 3.6395
            Mean Absolute Error (MAE) : 2.6454
          Mean Percentage Error (MPE) : -2.6958
Mean Absolute Percentage Error (MAPE) : 12.9926

Accuracy Measures for Validation Set - All Variables

Regression statistics

                      Mean Error (ME) : 0.2023
       Root Mean Squared Error (RMSE) : 3.8378
            Mean Absolute Error (MAE) : 2.8230
          Mean Percentage Error (MPE) : -4.5533
Mean Absolute Percentage Error (MAPE) : 14.6529
```

The Mean Absolute Percentage Error (MAPE) is a reliable performance metric to measure the accuracy of predictions, where a smaller percentage of error indicates a higher level of accuracy.

In this, Both the training and validation sets show comparable accuracy measures, suggesting no overfitting.

The MAPE value of the validation set is 14.65%, which is significantly low, indicating the model's robustness in making accurate predictions. Hence, we can conclude that overfitting is not a concern in this model.

3a. Use the Backward Elimination algorithm in Python to identify the best predictors for the multiple linear regression model. Based on these predictors, train a new multiple linear regression model using the respective training data set predictors and 80%-20% partition of the data set. Identify and display in Python the intercept and regression coefficients of this model and the common accuracy measures for validation partition. Provide these coefficients in your report and present the mathematical equation of the respective multiple linear regression model.

Intercept: 46.55

Regression Coefficients:

```
Regression Model for Training Set Using Backward Elimination

Intercept  46.55
       Predictor  Coefficient
0          CRIME        -0.13
1         INDUST         0.11
2      NIT_OXIDE       -17.47
3          ROOMS         0.61
4       DISTANCE        -0.73
5         RADIAL         0.20
6            TAX        -0.01
7       ST_RATIO        -0.59
8       LOW_STAT        -0.48
9     CHAR_RIV_Y         2.14
10   C_MVALUE_Yes        11.52
```

Regression Model Equation:

MVALUE = 46.55 -0.13 CRIME + 0.11 INDUST -17.47 NIT_OXIDE + 0.61 ROOMS -0.73 DISTANCE + 0.20 RADIAL -0.01 TAX -0.59 ST_RATIO -0.48 LOW_STAT + 2.14 CHAR_RIV_Y + 11.52 C_MVALUE_Yes

```
Accuracy Measures for Validation Set Using Backward Elimination

Regression statistics

                      Mean Error (ME) : 0.1904
       Root Mean Squared Error (RMSE) : 3.8356
            Mean Absolute Error (MAE) : 2.8137
          Mean Percentage Error (MPE) : -4.5767
Mean Absolute Percentage Error (MAPE) : 14.5939
```

As we can see from the above values that the MAPE is 14.59% which is slightly less indicating a better model with fewer variables.

3b. Use the Forward Selection algorithm in Python exactly as discussed in 3a. Provide the same results in your report as discussed in 3a. Also, explain if there are differences between the best predictors (number and specific predictors used) in the models in 3a and 3b.

Intercept: 46.55

Regression Coefficients:

```
Regression Model for Training Set Using Forward Selection

Intercept  46.55
       Predictor  Coefficient
0   C_MVALUE_Yes        11.52
1       LOW_STAT        -0.48
2          CRIME        -0.13
3     CHAR_RIV_Y         2.14
4       ST_RATIO        -0.59
5       DISTANCE        -0.73
6      NIT_OXIDE       -17.47
7         RADIAL         0.20
8            TAX        -0.01
9         INDUST         0.11
10         ROOMS         0.61
```

Regression Model Equation:

MVALUE = 46.55 -0.13 CRIME + 0.11 INDUST -17.47 NIT_OXIDE + 0.61 ROOMS -0.73 DISTANCE + 0.20 RADIAL -0.01 TAX -0.59 ST_RATIO -0.48 LOW_STAT + 2.14 CHAR_RIV_Y + 11.52 C_MVALUE_Yes

```
Accuracy Measures for Validation Set Using Forward Selection

Regression statistics

                      Mean Error (ME) : 0.1904
       Root Mean Squared Error (RMSE) : 3.8356
            Mean Absolute Error (MAE) : 2.8137
          Mean Percentage Error (MPE) : -4.5767
Mean Absolute Percentage Error (MAPE) : 14.5939
```

In both the 3a & 3b models the same number of predictors and exact same predictors are used.

3c. Use the Stepwise algorithm in Python exactly as discussed in 3a. Provide the same results in your report as discussed in 3a. Also, explain if there are differences between the best predictors (number and specific predictors used) in the models in 3b and 3c.

Intercept: 46.55

Regression Coefficients:

```
Regression Model for Training Set Using Stewise Selection

Intercept  46.55
       Predictor  Coefficient
0    C_MVALUE_Yes        11.52
1        LOW_STAT        -0.48
2           CRIME        -0.13
3      CHAR_RIV_Y         2.14
4        ST_RATIO        -0.59
5        DISTANCE        -0.73
6       NIT_OXIDE       -17.47
7          RADIAL         0.20
8             TAX        -0.01
9          INDUST         0.11
10          ROOMS         0.61
```

Regression Model Equation:

MVALUE = 46.55 -0.13 CRIME + 0.11 INDUST -17.47 NIT_OXIDE + 0.61 ROOMS -0.73 DISTANCE + 0.20 RADIAL -0.01 TAX -0.59 ST_RATIO -0.48 LOW_STAT + 2.14 CHAR_RIV_Y + 11.52 C_MVALUE_Yes

```
Accuracy Measures for Validation Set Using Stepwise Selection

Regression statistics

                      Mean Error (ME) : 0.1904
       Root Mean Squared Error (RMSE) : 3.8356
            Mean Absolute Error (MAE) : 2.8137
          Mean Percentage Error (MPE) : -4.5767
Mean Absolute Percentage Error (MAPE) : 14.5939
```

In both the 3b & 3c models the same number of predictors and exact same predictors are used.

3d. Present and compare in your report the common accuracy measures for the validation data set of the 4 linear regression models: with all predictors and based on the Backward Elimination, Forward Selection, and Stepwise algorithms. Using the value of RMSE and the number of variables in each model, which model would you recommend using for making predictions in this case? Briefly explain your answer.

	Mean Error (ME)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)	Mean Percentage Error (MPE)	Mean Absolute Percentage Error (MAPE)
Multiple linear regression model	0.2023	3.8378	2.8230	-4.5533	14.6529
Backward Elimination	0.1904	3.8356	2.8137	-4.5767	14.5939
Forward Selection	0.1904	3.8356	2.8137	-4.5767	14.5939
Stepwise algorithms	0.1904	3.8356	2.8137	-4.5767	14.5939

The Number of variables in the Multiple linear regression model is 13, whereas, the number of variables in Backward Elimination, Forward Selection, and Stepwise algorithms is 11.

The value of RMSE in the Multiple linear regression model is 3.8378, whereas, the value of RMSE in Backward Elimination, Forward Selection, and Stepwise algorithms is 3.8356. The values are similar.

The Model which would be recommended is either of Backward Elimination, Forward Selection, or Stepwise algorithms as the number of variables is less than the multiple linear regression model and RMSE is also slightly less.