FEATURE SELECTION

Feature selection is the process of selecting a subset of relevant features (variables, attributes) from a larger set of available features to use in a machine learning model. The goal of feature selection is to improve the performance of the model by reducing the number of irrelevant or redundant features, which can decrease model accuracy and increase training time.

The feature selection process typically involves the following steps:

1. **Define the problem**: The first step is to define the problem and the objective of the analysis. This includes identifying the variables that are relevant to the problem and determining the type of model that will be used.
2. **Data preprocessing**: This step involves cleaning and preprocessing the data. This includes handling missing values, dealing with outliers, and normalizing the data.
3. **Feature selection techniques**: There are several techniques that can be used for feature selection, including:
    a. Filter methods: These methods select features based on statistical measures such as correlation, mutual information, or chi-squared tests. These methods are computationally efficient but may not consider the interactions between features.
    b. Wrapper methods: These methods use a machine learning algorithm to evaluate the performance of different feature subsets. These methods can be computationally expensive but can consider the interactions between features.
    c. Embedded methods: These methods incorporate feature selection as part of the model building process. For example, regularization methods such as Lasso or Ridge regression can be used to penalize the coefficients of irrelevant features, effectively eliminating them from the model.
4. **Evaluation**: After selecting a subset of features, the performance of the model is evaluated using a validation set or cross-validation. The selected features may be re-evaluated and refined using the validation results.
5. **Model training**: Once the optimal subset of features has been selected, the final model is trained using the entire dataset.

It's important to note that feature selection is an iterative process that requires domain knowledge and expertise in order to select the most relevant and informative features for a given problem. Additionally, it's important to balance the trade-off between model performance and computational efficiency, as selecting too few or too many features can negatively impact the accuracy and efficiency of the model.

Multicollinearity and Variation Inflation Factor

The dependent variable should have a strong relationship with independent variables. However, any independent variables should not have a strong correlation among other independent variables. Multicollinearity is an incident where one or more of the independent variables are strongly correlated with each other. In such an incident, we should use only one among correlated independent variables. Multicollinearity and variance inflation factor (VIF) is an indicator of the existence of multicollinearity, and statsmodel provides a function to calculate the VIF for each independent variable' a value of greater than 10 is the rule of thumb for the possible existence of high multicollinearity. The standard guideline for VIF value is: VIF = 1 means no correlation exists, and VIF >1 but <5 means moderate correlation existing.

$$VIF_i = 1/(1-R_i^2)$$

where $R_i{}^2$ is the coefficient of determination of variable X_i

```
#####Multicollinearity and VIF#####
import pandas as pd
from sklearn import preprocessing
import statsmodels.graphics.api as smg
import matplotlib.pyplot as plt
# Load data
df = pd.read_csv('F:/Jan-May___2023/CS-317__slides/Python_Programs_tutorials/Practical__PYTHON/basic_materials/2__code__data/Chapter_3_Code/Code/Data/Housing_Modified.csv')
# Convert binary fields to numeric boolean fields
lb = preprocessing.LabelBinarizer()
df.driveway = lb.fit_transform(df.driveway)
df.recroom = lb.fit_transform(df.recroom)
df.fullbase = lb.fit_transform(df.fullbase)
df.gashw = lb.fit_transform(df.gashw)
df.airco = lb.fit_transform(df.airco)
df.prefarea = lb.fit_transform(df.prefarea)
# Create dummy variables for stories
df_stories = pd.get_dummies(df['stories'], prefix='stories', drop_first=True)
# Join the dummy variables to the main dataframe
df = pd.concat([df, df_stories], axis=1)
del df['stories']
# lets plot the correlation matrix using statmodels graphics packages' plot_corr
# create correlation matrix
corr = df.corr()
print(df)
smg.plot_corr(corr, xnames=list(corr.columns))
plt.show()
```

We can see from the plot that stories_one has a strong negative correlation with stories_two. Let’s perform the VIF analysis to eliminate strongly correlated independent variables

```
#####Remove Multicollinearity#####
import numpy as np
from statsmodels.stats.outliers_influence import variance_inflation_factor, OLSInfluence
# create a Python list of feature names
independent_variables = ['lotsize', 'bedrooms', 'bathrms','driveway', 'recroom',
 'fullbase','gashw','airco','garagepl', 'prefarea',
 'stories_one','stories_two','stories_three']
# use the list to select a subset from original DataFrame
X = df[independent_variables]
y = df['price']
thresh = 10
for i in np.arange(0,len(independent_variables)):
    vif = [variance_inflation_factor(X[independent_variables].values, ix) for ix in range(X[independent_variables].shape[1])]
    maxloc = vif.index(max(vif))
    if max(vif) > thresh:
        print("vif :", vif)
        print('dropping \'' + X[independent_variables].columns[maxloc] + '\'at index: ' + str(maxloc))
        del independent_variables[maxloc]
    else:
        break
print('Final variables:', independent_variables)
```

We can see that VIF analysis has eliminated bedrooms greater than 10; however, stories_one and stories_two have been retained. Let's run the first iteration of a multivariate regression model with the set of independent variables that have passed the VIF analysis. To test the model performance, the common practice is to split the data set into 80/20 (or 70/30) for train/test, respectively, and use the train data set to build the model. Then apply the trained model on the test data set to evaluate the performance of the model.

```
#####Build the Multivariate Linear Regression Model#####
from sklearn.model_selection import train_test_split
from sklearn import metrics
import statsmodels.api as sm
# create a Python list of feature names
independent_variables = ['lotsize', 'bathrms','driveway',
'fullbase','gashw', 'airco','garagepl',
 'prefarea','stories_one','stories_three']
# use the list to select a subset from original DataFrame
X = df[independent_variables]
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=.80,
random_state=1)
# create a fitted model
lm = sm.OLS(y_train, X_train).fit()
# print the summary
print(lm.summary())
# make predictions on the testing set
y_train_pred = lm.predict(X_train)
y_test_pred = lm.predict(X_test)
y_pred = lm.predict(X) # full data
```

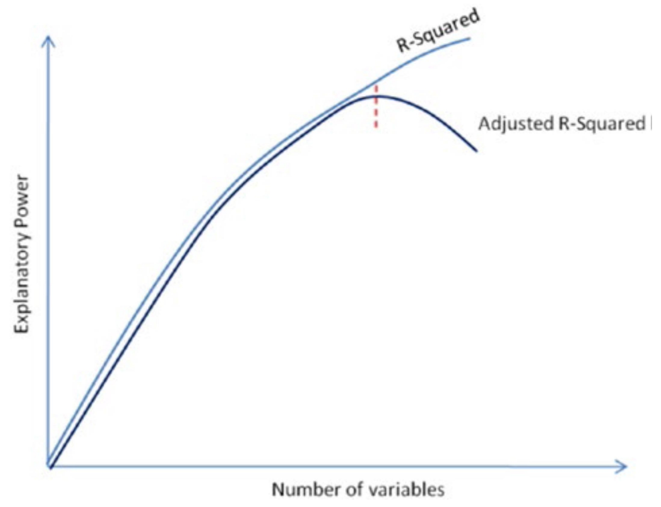
```
print("Train MAE: ", metrics.mean_absolute_error(y_train, y_train_pred))
print("Train RMSE: ", np.sqrt(metrics.mean_squared_error(y_train,
y_train_pred)))
print("Test MAE: ", metrics.mean_absolute_error(y_test, y_test_pred))
print("Test RMSE: ", np.sqrt(metrics.mean_squared_error(y_test,
y_test_pred)))
```

Interpreting the Ordinary Least Squares (OLS) Regression Results

Adjusted R-Squared: The simple R-Squared value will keep increasing with the addition of an independent variable. To fix this issue, adjusted R-Squared is considered for multivariate regression to understand the explanatory power of the independent variables.

$$\text{Adjusted}\, R^2 = 1 - \frac{\left(1 - R^2\right)\left(N - 1\right)}{N - p - 1}$$

where N is total observations or sample size and p is the number of predictors.



R-Squared vs. Adjusted R-Squared

— Figure given above shows how R-Squared follows Adjusted R-Squared with increase of more variables
— With inclusion of more variables R-Squared always tend to increase
— Adjusted R-Squared will drop if the variable added does not explain the variable in the dependent variable

Coefficient: This is the individual coefficient for the respective independent variables.
It can be either a positive or a negative number, which indicates that an increase in every unit of that independent variable will have a positive or negative impact on the dependent variable value.

Standard error: This is the average distance of the respective independent observed values from the regression line. The smaller values show that the model fitting is good. Durbin-Watson: It's one of the common statistics used to determine the existence of multicollinearity, which means two or more independent variables used in the multivariate regression model are highly correlated.

The **Durbin-Watson statistic** is always a number between 0 and 4. A value around 2 is ideal (range of 1.5 to 2.5 is relatively normal); it means that there is no autocorrelation between the variables used in the model.

Confidence interval: This is the coefficient to calculate a 95% confidence interval for the independent variable's slope.

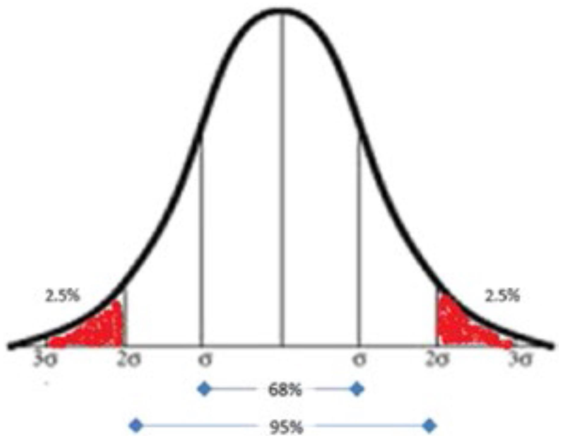
t and p-value: p-value is one of the important statistics. In order to better understand, we'll have to explore the concept of hypothesis testing and normal distribution.

<u>HYPOTHESIS TESTING</u>

Hypothesis testing is an assertion regarding the distribution of the observations and validating this assertion. The hypothesis testing steps are as follows:

- A hypothesis is made.
- The validity of the hypothesis is tested.
- If the hypothesis is found to be true, it is accepted.
- If it is found to be untrue, it is rejected
- The hypothesis that is being tested for possible rejection is called the null hypothesis.
- The null hypothesis is denoted by H0.
- The hypothesis that is accepted when the null hypothesis is rejected is called an alternate hypothesis Ha.
- The alternative hypothesis is often the interesting one and often the one that someone sets out to prove.
- For example, null hypothesis H0 is that the lot size has a real effect on house price; in this case, the coefficient m is equal to zero in the regression equation ($y = m * \text{lot size} + c$).
- Alternative hypothesis Ha is that the lot size does not have a real effect on house price, and the effect you saw was due to chance, which means the coefficient m is not equal to zero in the regression equation.

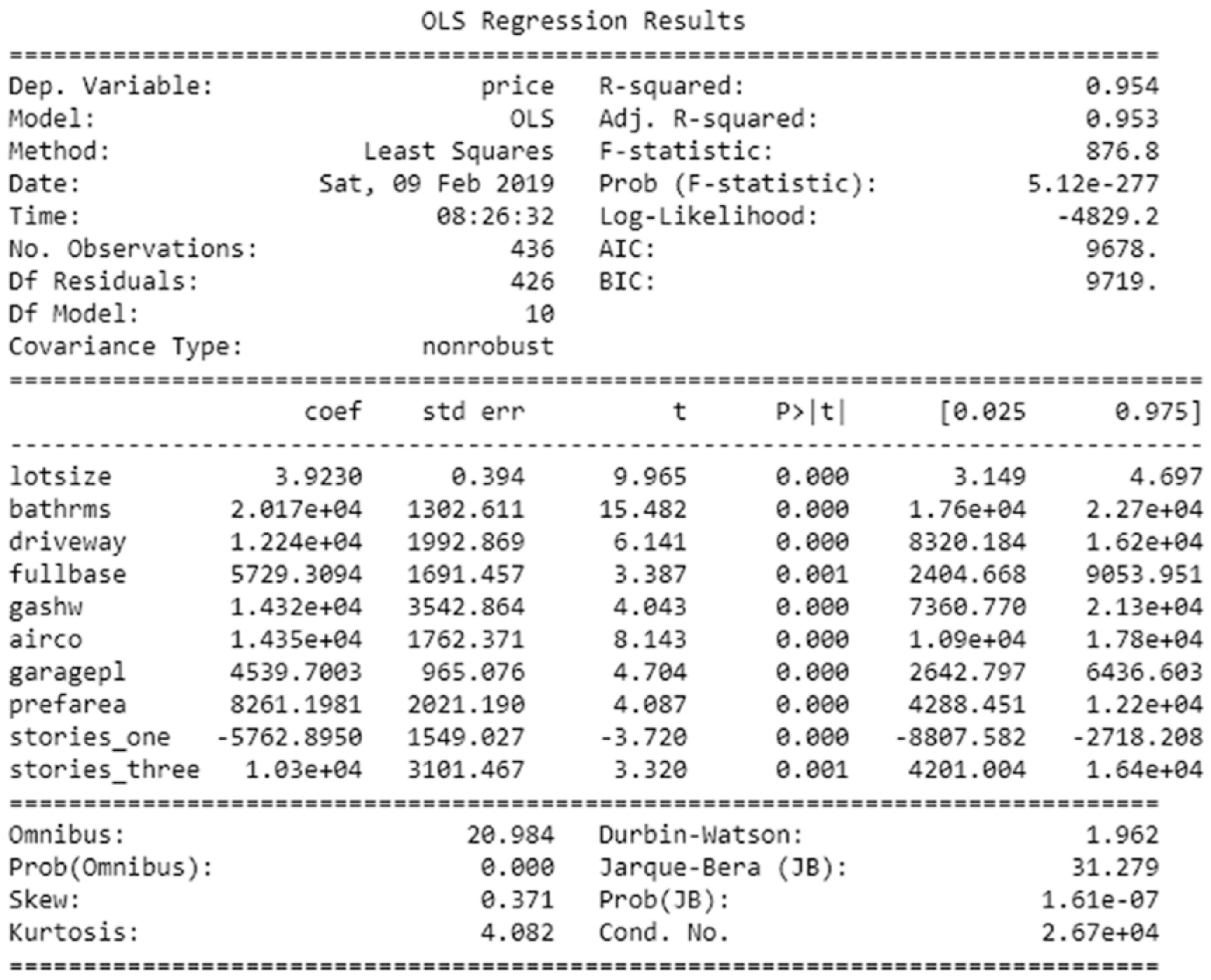
• In order to be able to say whether the regression estimate is close enough to the hypothesized value to be acceptable, we take the range of estimate implied by the estimated variance and see whether this range will contain the hypothesized value. To do this, we can transform the estimate into a standard normal distribution, and we know that 95% of all values of a variable that has a mean of 0 and variance of 1 will lie within 0 to 2 standard deviation. Given a regression estimate and its standard error, we can be 95% confident that the true (unknown) value of m will lie in this region.



Normal distribution (red is the rejection region)

• The t-value is used to determine a p-value (probability), and p-value ≤ 0.05 signifies strong evidence against the null hypothesis, so you reject the null hypothesis. A p-value >0.05 signifies weak evidence against the null hypothesis, so you fail to reject the null hypothesis. So in our case, the variables with ≤ 0.05 means variables are significant for the model.

- The process of testing a hypothesis indicates that there is a possibility of making an error. There are two types of errors for any given data set, and these two types of errors are inversely related, which means the smaller the risk of one, the higher the risk of the other.
- Type I error: The error of rejecting the null hypothesis H0 even though H0 was true
- Type II error: The error of accepting the null hypothesis H0 even though H0 was false
- Note that variables “stories_three” and “recroom” have a large p-value, indicating it’s insignificant.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.954
Model:                            OLS   Adj. R-squared:                  0.953
Method:                 Least Squares   F-statistic:                     876.8
Date:                Sat, 09 Feb 2019   Prob (F-statistic):          5.12e-277
Time:                        08:26:32   Log-Likelihood:                -4829.2
No. Observations:                 436   AIC:                             9678.
Df Residuals:                     426   BIC:                             9719.
Df Model:                          10
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
lotsize           3.9230      0.394      9.965      0.000       3.149       4.697
bathrms        2.017e+04   1302.611     15.482      0.000    1.76e+04    2.27e+04
driveway       1.224e+04   1992.869      6.141      0.000    8320.184    1.62e+04
fullbase       5729.3094   1691.457      3.387      0.001    2404.668    9053.951
gashw          1.432e+04   3542.864      4.043      0.000    7360.770    2.13e+04
airco          1.435e+04   1762.371      8.143      0.000    1.09e+04    1.78e+04
garagepl       4539.7003    965.076      4.704      0.000    2642.797    6436.603
prefarea       8261.1981   2021.190      4.087      0.000    4288.451    1.22e+04
stories_one   -5762.8950   1549.027     -3.720      0.000   -8807.582   -2718.208
stories_three   1.03e+04   3101.467      3.320      0.001    4201.004    1.64e+04
==============================================================================
Omnibus:                       20.984   Durbin-Watson:                   1.962
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               31.279
Skew:                           0.371   Prob(JB):                     1.61e-07
Kurtosis:                       4.082   Cond. No.                     2.67e+04
==============================================================================
```

Train MAE: 11993.3436816

Train RMSE: 15634.9995429

Test MAE: 12902.4799591

Test RMSE: 17694.9341405

- Note that dropping the variables has not impacted adjusted R-Squared negatively.