Exercise 4 : Linear Regression

Workflow

1. Download the .ipynb files and data files posted with this exercise and store them all in a folder on your Desktop.
2. Open Jupyter Notebook (already installed on the Lab computers) and navigate to the aforesaid folder on Desktop.
3. Open and explore the .ipynb files (notebooks) that you downloaded, and go through "Preparation", as follows.
4. The walk-through videos posted on NTU Learn (under Course Content) may help you with this "Preparation" too.
5. Create a new Jupyter Notebook, name it Exercise4_solution.ipynb, and save it in the same folder on the Desktop.
6. Solve the "Problems" posted below by writing code, and corresponding comments, in Exercise4_solution.ipynb.

Try to solve the problems on your own. Take help and hints from the "Preparation" codes and the walk-through videos. **If you are still stuck, talk to your friends in the Lab to get help/hints.** If that fails too, approach your Lab Instructor.

Note : Don't forget to import the Essential Python Libraries required for solving the Exercise. Write code in the usual "Code" cells, and notes/comments in "Markdown" cells of the Notebook. Check the preparation notebooks for guidance.

Preparation

M3 LinearRegression.ipynb	Check how to perform Linear Regression on the Pokemon data (pokemonData.csv)

Objective

In the last Lab Exercise, you have identified and analyzed some of the most relevant numeric variables in this dataset, which may affect the sale price of a house, and hence, will probably be most relevant in predicting "SalePrice". In this Lab Exercise, you will utilize some of those numeric variables to perform Linear Regression and predict "SalePrice".

Typical steps to follow while building a supervised machine learning model on a given dataset:

- **Partition** the labeled dataset into two random portions – 80% to Train the model and 20% to Test the model.
- **Fit** the desired supervised machine learning model on the Train set to predict response using the predictors.
- **Predict** response using the predictors on the Test set using the machine learning model fit on the Train data.
- **Check** the Goodness of Fit of the model on Train set using R^2 and Prediction Accuracy on Test set using MSE.

Disclaimer: There may be several ways to solve these problems and there is no single correct answer. Try to explore on your own, talk to your friends and the Lab Instructor, and make sure you are happy with your own justifications. You will get marks for your solutions as long as your justifications make sense, and you can explain those clearly.

Marks distribution

3 points for Problem 1	2 points for train-test set and regression in (a) + 1 point for metrics in (b)
3 points for Problem 2	2 points for the two regressions + 1 point for comparison and justifications
4 points for Problem 3	2 points for outlier removal in (a) + 1 point for model in (b) + 1 point for (c)

Problems

Problem 1 : Predicting SalePrice using GrLivArea

Note : We observed during EDA that GrLivArea and SalePrice have a strong linear relationship with correlation 0.71. In this problem, you will build a Linear Regression model to predict SalePrice using GrLivArea and judge its accuracy.

a) Create appropriate datasets for Train and Test in an 80:20 ratio and fit a Linear Regression model on the Train set to predict SalePrice using GrLivArea. Print the coefficients of your model and plot the regression line.

b) Check the Goodness of Fit of the model on the Train set and Prediction Accuracy of the model on the Test set. Print the *metrics* for Goodness of Fit and Prediction Accuracy that you think are appropriate in each scenario.

Hints and Pointers

- If you take just the first 80% of the data as train and the next 20% as test, it may not be the best train test split.
- If you obtain the coefficients for the regression line (intercept and coef), it should be easy to plot the line too.
- Goodness of Fit on train depends on the variance you explain, while prediction accuracy depends on the errors.

Problem 2 : Predicting SalePrice using Other Variables

Following the steps from the previous problem, build two new uni-variate Linear Regression models to predict SalePrice using the variables TotalBsmtSF and GarageArea, individually. Justify which of the three models is the best in this case.

Hints and Pointers

- Same as Problem 1, just on other variables. You can compare models using the metrics you are printing anyway.
- Optional: You may think of writing a simple Python function to do regression on some variables in a given dataset.

Problem 3 : Refining the Models to Predict SalePrice

In this problem, you will consider finer details of the dataset and the variables to refine the model to predict SalePrice.

(a) Find the houses (rows) that are “outliers” for GrLivArea and/or SalePrice. This means outliers for GrLivArea UNION outliers for SalePrice in a set notation. Remove all these “outliers” from the dataset so that it is clean.

(b) In the outlier-free dataset, create Train and Test sets with an 80:20 ratio, and fit a Linear Regression model on the Train set to predict SalePrice using GrLivArea. Print the model coefficients and plot the regression line.

(c) Check the Goodness of Fit of the model on the Train set and Prediction Accuracy of the model on the Test set. Do you think this model is better than the model obtained in Problem 1 for the same variables? Briefly justify.

Hints and Pointers

- In Part (a), there are houses that are outliers on both variables (intersection), but we really want the “union”.
- Part (b) is the same as Problem 1, just on the new outlier-free dataset. Check that the data now has less rows.
- Part (c) is the same as Problem 2, but keep in mind that the dataset has changed in this case, not the variables.

Hints are not meant to tell you exactly what to do for the problems; use these as pointers to search online. Take a close look at **Pandas DataFrame documentation** and check **Linear Regression LAMS** carefully to solve most of these problems.

LinearRegression : https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html