EECS461/ECE523
MACHINE LEARNING
Fall 2018

ASSIGNMENT 1

Due Date: *Sunday, November 4th, 2018, 23:59*

Assignment Submission: *Turn in your assignment by the due date through LMS. Prepare a single Jupyter Notebook (.ipynb) with the answers to all questions.* ***Name the file as <your first name>_<your last name>_ assignment1.ipynb.*** *Make sure to* ***use the sample Jupyter Notebook file provided to you as template****.*

All work in questions must be your own; you must neither copy from nor provide assistance to anybody else*. If you need guidance for any question, talk to the instructor or TA in office hours. You can also reach to TA Hacer Tilbeç at hacertilbec@std.sehir.edu.tr*

Late Assignment Policy: You have a total of ***4 days of late assignment*** turn-in allowance throughout this semester. For a single assignment, you can use ***a maximum of 2 late-days.*** You decide which assignments you are going to use your 4 late-days. After assignment due date/time, each 24-hours period is counted as one late date (i.e., if you submit late 1 hour or 23 hours, you use 1 late-date). It is your responsibility to keep track of your late days. If you are late more than 2 days for any assignment or you exhausted your late days, you get 0 from the late assignment (No exceptions)

In this assignment, you will do exploratory data analysis to understand a dataset and its features, do data processing and preparation, apply machine learning methods on this data to generate regression and classification models, analyze your models and generate predictions using the models. This assignment is mainly about the examples in the chapter 2 and 3 of the course book with a different data set. Reviewing the book and the corresponding code will greatly help you. ***You are expected to primarily use Scikit-Learn library in the assignment.***

DATA SET

With a dataset consists of features describing various aspects of residential homes in Ames, Iowa, this assignment challenges you to ***predict the sale price of each home***. Data is in CSV format and has already been split into training and test sets for your convenience.

train.csv: the training set

test.csv: the test set

DATA EXPLORATION & PREPARATION (30 points)

In the first part of the assignment, you will analyze the dataset and make changes in it in order to prepare the data for machine learning algorithms.

(a) (5 points) You are going to use "SalePrice" column as the label (target that you are going to predict). For this reason, create two pandas dataframes by using train.csv, one will contain the input features and the other will contain only the label. Name these dataframes as **train_x_a** and **train_y** respectively.

(b) (5 points) Find all features (columns) that contain missing (NaN) values in it and store these features' names in **nan_columns** list. Fill the missing values with the median value of the corresponding feature. Your new data frame without any missing values should be named as **train_x_b.** (Note that if there is any missing values in the label - SalePrice column-, drop the corresponding row completely from train_x and train_y)

(c) (5 points) Find features (columns) that have categorical values (strings) in it and store these column names in **categorical_columns** list.
For example, if data has 'gender' column with 'female' and 'male' values, then 'gender' should be in the **categorical_columns** list.

(d) (10 points) Perform ***one hot encoding*** on features with categorical values. Modify train_x_b by replacing categorical columns with their one-hot encoding representations. Name your modified dataframe as **train_x_d .**
For example, if you have a column named "gender" that has two unique values (male and female), after one-hot encoding, gender column will be replaced with two new columns in the dataframe, one column for male and one column for female. If your original sample has female value as gender, then your new dataframe will have 1 in the female column and 0 in the male column.

(e) (5 points) Scale all columns in train_x_d with standardization. Name the new dataframe with scaled values as **train_x_e**.
Note: Scikit-Learn provides a transformer called StandardScaler for standardization. The output of the scaler is a numpy array. You need to convert it dataframe after standardization. Don't forget to add indexes and columns of the original dataframe to the new dataframe.

LINEAR REGRESSION TO PREDICT HOUSE PRICES (35 points)

In this part of the assignment, you are going to train a Linear Regression model that predicts the sale prices of houses by using the other features in the dataset.

(f) (5 points) Create a Linear Regression model with default parameters. Train the model

with the train_x_e and train_y. Print the Mean Square Error (MSE) for training data.

(g) (10 points) Perform 5-fold cross validation with training data and print MSE score for each fold and the their average (mean)

(h) (10 points) Using test.csv, create **test_x** that as all the features except SalePrice and **test_y** that has ScalePrice. Fill missing values in test_x with median value, perform feature scaling and apply one hot encoding to categorical values, same as what you did in the first part of the assignment.
Note that transformations you apply to the test set should be same as the one applied to training set. When filling the missing values, median value should be the one computed for the training set. Similarly, in standardization, mean and standard deviation should be the ones from training set.

(i) (10 points) Predict sale prices of houses in ***test_x*** data using your linear regression model. Store predicted values in **predicted_values** variable. Print test MSE of your model.

CLASSIFICATION MODEL TO PREDICT HOUSE PRICE CATEGORY (35 points)

In this part, you will use the same dataset above but you will use another approach to predict price of a house. However, this time, you will not predict an exact price but you will try to predict a price category for each house such as cheap, expensive and luxury. You looked at the prices and decided to create 5 different house price categories as discussed below. You label these categories simply with integers between 1 and 5. Consequently, you decide to develop a multiclass classification model. Your model will predict the price category (1,2,3,4 or 5) based on the input features. Give it a try and see what kind of accuracy you get.

(j) (10 points) You will segment house prices into 5 categories. Use following rules for house price categories to transform house prices in train_y and test_y into the corresponding categories. Name new dataframes as train_y_j and test_y_j. These will be the labels that you will predict with your classification model.
You don't need to make any additional changes in your feature values. You can use train_x_e and test_x that you created in part (e) and part (h) after various transformations

house price < 100.000	→	label=1
100.000 <= house price < 200.000	→	label=2
200.000 <= house price < 300.000	→	label=3
300.000 <= house price < 400.000	→	label=4
400.000 <= house price	→	label=5

(k) (5 points) Train a multiclass classification model (you can use any one from chapter 3 exercises with default parameters).

(l) (10 points) Perform 5-fold cross validation with training data and calculate confusion matrix, accuracy, precision, recall and f1 scores (Note: For precision, recall and f1 score calculations, set *average* parameter to *'micro'*).

(m) (10 points) Predict sale price category of houses in test data ***test_x*** using your multiclass classification model. Store predicted values in **predicted_values** variable. Calculate confusion matrix, accuracy, precision, recall and f1 scores of your model (Note: For precision, recall and f1 score calculations, set *average* parameter to *'micro'*).

IMPORTANT NOTES

- Prepare and upload one Jupyter notebook file, which should be named as <your first name>_<your last name>_ assignment1.ipynb.
- A template Jupyter notebook file provided to you. Follow the template's structure.
- Explain your code with comments.

Wrong file name format	-10 points
Not using template	-10 points
Not using correct variable (dataframe) names	-10 points
Insufficient comment	-10 points