COMP 468 Machine Learning in Python, Spring 2023
Instructor: Zafer Aydın
Homework 2

Setup

1. Login to your Kaggle account.
2. Click on the Code link.
3. Click on the New Notebook button.
4. Change the title of the page on the upper left corner so that it obeys this format: "ML in Python, Spring 2023, Homework 2, Your Name Surname" (e.g. ML in Python, Spring 2023, Homework 2, Zafer Aydın).
5. Click on the "Add Data" button on the upper right corner.
6. Click on the "Search keyword or URL text box" below Add Data. Search for "Housing Prices Competition for Kaggle Learn Users" by entering this text to the search box.
7. Click on the + button next to the first dataset found to add dataset.
8. Click on the × button next to Add Data at the upper right corner to close the window for adding data.
9. Click on the +Code button at the lower left side of the code cell to add a new code cell.
10. You can start from the template code called ML in Python, Spring 2023, Homework 2, Template (which is made available as ml-in-python-spring-2023-homework-2-template.ipynb).

Assignment

In this homework, you will work with numeric features only. The code template selects numeric features already. It also splits the original training set (i.e. X and y) into a new training set (X_train and y_train) and a validation set (X_valid and y_valid). In this homework, you will implement the three techniques we learned for handling missing values and compute performance on validation set as well as the test set (i.e. by submitting your test set predictions to leaderboard of the competition).

1. Drop the feature columns with missing values from X_train and X_valid, which is the first technique for handling missing values. Set the names of the new data frames after dropping columns as reduced_X_train and reduced_X_valid. Compute and print the MAE on validation set by calling the score_dataset function given in the code template.

2. Fill the missing values in X_train and X_valid using SimpleImputer, which is the second technique for handling missing values. Set the strategy argument of SimpleImputer to 'median'. Set the names of the new data frames after imputation as imputed_X_train and imputed_X_valid. Transfer column names from X_train to imputed_X_train and from X_valid to imputed_X_valid. Compute and print the MAE on validation set by calling the score_dataset function given in the code template.

3. Implement the third strategy we learned for handling missing values on tutorial page of Lesson 2 of Intermediate Machine Learning course. For this purpose, for each feature column with missing values, insert a new feature column so that the value at a particular row of this new column is True if the data at the same row is missing on the original feature column and False if the data at the same row is non-missing on the original feature column. Set the names of the new data frames after adding the new feature columns (with True/False values) as X_train_plus and X_valid_plus. Apply SimpleImputer after adding new columns for columns with missing values by setting the strategy argument to 'median'. Set the names of the new data frames after imputation as imputed_X_train_plus and imputed_X_valid_plus. Transfer column names from X_train_plus to imputed_X_train_plus and from X_valid_plus to

imputed_X_valid_plus. Compute and print the MAE on validation set by calling the score_dataset function given in the code template.

4. Repeat question 1 this time starting from X (i.e. original training set) to find columns with missing values. Drop the columns with missing values from X and X_test. Set the names of the new data frames after dropping columns as reduced_X and reduced_X_test. After dropping columns from X_test, it may still contain some other columns with missing values but these columns are not among the ones we removed (i.e. those are columns with missing values in test set but not in training set). To solve this problem, one approach is to apply imputation on the reduced versions of training and test sets. For this purpose, apply SimpleImputer on reduced_X and reduced_X_test by setting the strategy argument to 'median'. Set the names of the new data frames after imputation as reduced_imputed_X and reduced_imputed_X_test. Transfer column names from reduced_X to reduced_imputed_X and from reduced_X_test to reduced_imputed_X_test. Train a random forest model with n_estimators set to 100 and random_state to 0 using training set (i.e. reduced_imputed_X). Compute predictions on the test set (i.e. reduced_imputed_X_test). Save your predictions as a pandas data frame and convert it to a csv file as we did in class. Set the name of the csv file as submission_drop_columns.csv. Follow the steps below to submit your predictions to competition.

- Run your notebook
- Once it finishes running, click on the /kaggle/working directory under the Output section at the right panel. Right next to submission_drop_columns.csv click on the vertical three dots and click on the Download button.
- Go to https://www.kaggle.com/competitions/home-data-for-ml-course link. Make sure you first register to the competition first if you haven't done yet. Click on the Submit Predictions button. Click on the Browse Files button and submit the prediction file you downloaded to the challenge.
- Go back to your notebook and add a Markdown text cell after the latest code cell by clicking on the +Markdown button at the bottom. Enter your leaderboard score to this cell.

5. Repeat question 2, this time imputing missing values in X and X_test. Set the names of the new data frames after imputation as imputed_X and imputed_X_test. Transfer column names from X to imputed_X and from X_test to imputed_X_test. Train a random forest model with n_estimators set to 100 and random_state to 0 using training set (i.e. imputed_X). Compute predictions on the test set (i.e. imputed_X_test). Save your predictions as a pandas data frame and convert it to a csv file as we did in class. Set the name of the csv file as submission_impute_median.csv. Follow the steps as in question 4 to submit your predictions to competition.

6. Repeat question 3, this time starting from X and X_test. Set the names of the new data frames after adding the new feature columns (with True/False values) as X_plus and X_test_plus. Apply SimpleImputer after adding new columns for columns with missing values by setting the strategy argument to 'median'. Set the names of the new data frames after imputation as imputed_X_plus and imputed_X_test_plus. Transfer column names from X_plus to imputed_X_plus and from X_test_plus to imputed_X_test_plus. Train a random forest model with n_estimators set to 100 and random_state to 0 using training set (i.e. imputed_X_plus). Compute predictions on the test set (i.e. imputed_X_test_plus). Save your predictions as a pandas data frame and convert it to a csv file as we did in class. Set the name

of the csv file as submission_impute_plus_median.csv. Follow the steps as in question 4 to submit your predictions to competition.

7. Fill the table below that includes your validation set and test set scores.

	Validation Set MAE	Test Set MAE
Drop columns with missing values		
SimpleImputer, strategy=median		
Add new column, SimpleImputer, strategy=median		

8. Which approach gives the best validation set score? Which approach gives the best test set score (i.e. leaderboard score)? If the approach that gives the best validation score is different from the approach that gives the best test set score, what could be the reason for this behavior?

Submission

Once you finish, click File and Download Notebook. Submit your notebook with .ipynb extension to Canvas. Your notebook should include all the codes you developed for each question. Implement each question as a separate and a single code cell and put a comment line that includes the question number at the beginning of the code cell of each question (e.g. #Question 1). Submit your answers to questions 7 and 8 as a text or Word document to Canvas.