INFO 1998 Project B Deliverable

Linear Regression and KNN Classifications

Guideline and Rubric

Release Date: March 7th
Due Date: March 20th at 11:59 pm
Submit Through: CMS

Overview

We have now begun to learn about different types of models in lecture; specifically, we have already covered both linear (continuous data) regression and KNN (categorical data) classification. Remember KNN classification can be performed on binned numerical data. You will be using the same train.csv from the housing dataset for this project as well. Since the projects are building on each other, we encourage you to use **your modified csv from the last project**. A major portion of this assignment is feature engineering! So make sure you are putting time and effort into picking your features or your models will not be very good.

For the **linear regression model** component, you will be predicting the **housing price**. There are no restrictions on which columns can be used as features or how many features your model can have. Therefore, you should use correlation, plotting, feature engineering, or anything else you can think of to decide on the best features for your model. In order to ensure that you have a decent understanding of a linear model, your predictions must be 40% better the baseline. The baseline is calculated by using RMSLE or RMSE and in the case of linear regression, your model's RMSLE/ RMSE must be 40% **lower** than the baseline. The code should contain **five different train/ test splits** that all are 40% more accurate than the baseline.

For the **KNN model** component, you will be predicting **pre1970 (excluding 1970) and post1970 in the YearBuilt column**. Note the predicted column is a binary column with a value of either 0 or 1 (or False / True, whichever one you prefer). Again, there are no restrictions on the columns used as features (although you cannot use YearBuilt); feature engineering is encouraged. This model must also be 50% more accurate than the baseline. The baseline is calculated differently for KNN classification; it is equivalent to the percentage of the column's most frequent value. Your predicted column's accuracy must be 50% **above** the baseline. Same as the linear model, you should create **five different train/ test splits** that all are 50% more accurate than the baseline.

Finally, you will submit a **brief** paragraph on each of your models. This paragraph should discuss how your features were chosen as well as the performance of your model.

What to Submit:

A jupyter notebook containing

- any code you used to decide upon features (optional)
- your linear regression model code
- 5 different train/ test splits (you need to include 5 different **random_state** values you used) and their accuracy compared to the baseline
- your KNN classification model code
- 5 different train/ test splits (with 5 different **random_state** values you used) and their accuracy compared to the baseline
- one paragraph on your linear regression model (can be in a pdf)
- one paragraph on your KNN classification model (can be in a pdf)

The CSV you created in the previous project and imported for this one. If you don't submit one we will assume you just used the original train.csv.

Rubric

Criteria	Points
Linear Regression	
Correct Model - Model is linear with features	5
Correct Formula - Split baseline and prediction accuracy calculation	5
Accuracy: - Split is 40% more accurate (for all 5 train_test_split's) - (40% = 10/10, 36% ~ 9/10, 32% ~ 8/10, and so on)	10
Paragraph explanation	5
KNN Classification	
Correct Model - Model is classifying based on n_nearest neighbors	5
Correct Formula - Split baseline and prediction accuracy calculation	5
Accuracy: - Split is 50% more accurate (for all 5 train_test_split's) - (50% = 10/10, 45% ~ 9/10, 40% ~ 8/10, and so on)	10
Paragraph explanation	5
Total	**50**

Note the accuracy improvement calculation should be different for linear regression and kNN (because for linear regression you want to **reduce** RMSE and for kNN you want to **increase** the number of times your predicted and actual match).

linear regression: (baseline - model_accuracy) / baseline
kNN: (model_accuracy - baseline) / baseline