Task 3:

We used a new way to impute missing data in this step. We implemented imputation based on random forest regression and classification models, and call it rfimpute. This method is based on lecture notes about “Model driven imputation w/ RF”, but we modified it to incorporate categorical features.

rfimpute:

We start with a data frame where missing values are temporarily filled with mean(for numerical features) or mode(for categorical features).

For each feature column, if it does not have missing values in the original data frame, we keep this column unchanged(i.e. we do not need to do imputation for this column). If it has some missing values, then we form a training data frame using those rows having values in this feature column, and form a test data frame using those rows with missing values in this feature column. We do one hot encoding for both data frames. If the feature we are imputing is numeric, we train a random forest regression model on the training data frame(y is the feature column we are imputing, and X is the data frame of all other columns) , and predict on the test data frame. The predicted values are used to update the temporarily filled mean. If the feature we are imputing is categorical, we train a random forest classification model on the training data frame, and predict on the test data frame. The predicted labels are used to update the temporarily filled mode.

We loop through the above step up to 10 times. After each loop, we check whether the imputed data has significant change. If so, we keep updating the missing entries, if not, it means the imputation is good enough and we stop the loops and return.

We used this imputation function(rfimpute) to impute training data set and test data set separately, so that imputation will not leak information.

Features used under this task:

The features we used under this task are those features with less than 80% missing data, and are not description of another column(except special cases). We used one hot encoding to transform categorical features. We will investigate relative feature importances for our best model, and drop irrelevant columns in the next task (task 4).

Random Forest:

With this new way of imputation, we tried a random forest regression model. We used grid search to look for best parameters, and it turns out max_depth = 8 and n_estimators = 14 is the best.

With the random forest estimator using best parameters, our cross-validated score on training set is 0.9672.

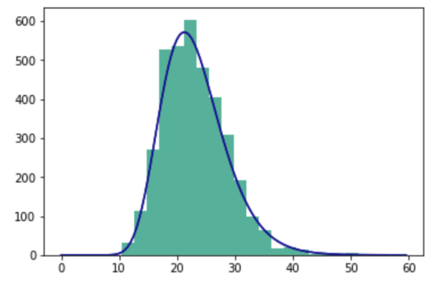
Gradient Boosting:

With the new way of imputation, we also tried a gradient boosting method. We also used grid search. The best parameters are: {'n_estimators': 70, 'max_depth': 21, 'min_samples_split': 250, 'max_features': 32}.

With the gradient boosting method and its best parameters, our cross-validated score on training set is 0.9597.

Transforming y:

From the historgram of the response variable y, we can see that the response variable is right skewed. By over-plotting a log-norm density, we can see that they fit very well. Therefore, we log-transformed y to see if we can have better predictive power.



Random Forest w/ log-transformed y:
Best parameters: {'n_estimators': 8, 'max_depth': 8}
cross-validated score on training set: 0.9702.

Gradient Boosting w/ log-transformed y:
Best parameters: {'n_estimators': 70, 'max_depth': 21, 'min_samples_split': 250, 'max_features': 32}.
cross-validated score on training set: 0.9650

Task 4:

Best Model:
From the models we tried in this task, we chose random forest w/ log-transformed y as our best model based on the results above. By applying it on test set, we got **test set score of 0.9467**. Some of the most important features for this model are as follows:

Importance Rank	Column name
1	Eng Displ
2	Annual Fuel1 Cost - Conventional Fuel
3	GHG Rating (1-10 rating on Label)_5
4	GHG Rating (1-10 rating on Label)_4
5	GHG Rating (1-10 rating on Label)_6
6	GHG Rating (1-10 rating on Label)_3
7	GHG Rating (1-10 rating on Label)_1
8	# Cyl
9	GHG Rating (1-10 rating on Label)_7

Importance Rank	Column name
10	GHG Rating (1-10 rating on Label)_2
11	GHG Rating (1-10 rating on Label)_10
12	Stop/Start System (Engine Management System) Code_Y
13	Model Year
14	GHG Rating (1-10 rating on Label)_8
15	GHG Rating (1-10 rating on Label)_9
16	Fuel Usage - Conventional Fuel_G
17	Fuel Usage - Conventional Fuel_DU
18	Carline Class
19	Index (Model Type Index)
20	Var Valve Lift?_Y
21	Var Valve Timing Desc_Variable Valve Timing System with inlet
22	Fuel Metering Sys Cd_CRDI
23	Var Valve Timing?_Y
24	Var Valve Lift?_N
25	Unique Label?_N
26	Max Ethanol % - Gasoline
27	Fuel Usage - Conventional Fuel_GP
28	# Gears
29	Fuel Usage - Conventional Fuel_GPR
30	Var Valve Timing Desc_variable valve timing at inlet and outlet valves

We then chose 30 most important features to keep, and drop the rest of the features.

By running the model again using this reduced dataset and the best parameters, we have:

Random Forest w/ log-transformed y and remove irrelevant features:
Test score: 0.9550

Therefore, by keeping only 30 most important features and remove other irrelevant features, there is some improvement.