Project 4: Regression Analysis

Lumi Huang, Christian Warloe, Ke Zhao, Landi Luo

Introduction

Regression analysis is a statistical procedure for evaluating whether or not there is a association between a outcome variables and a set of predictor variables, and for identifying the important predictors to include in the model. In this project, we will explore some basic regression models such as linear regression and neural network regression, and the techniques to avoid over-fitting. Cross-Validation and regularization are the two common techniques to handle over fittings. Cross-Validation evaluates models by training one or more models on subsets of the input data and testing on the complementary subset of the data. Regularization, on the other hand, penalizes on complex models by penalizing the loss function.

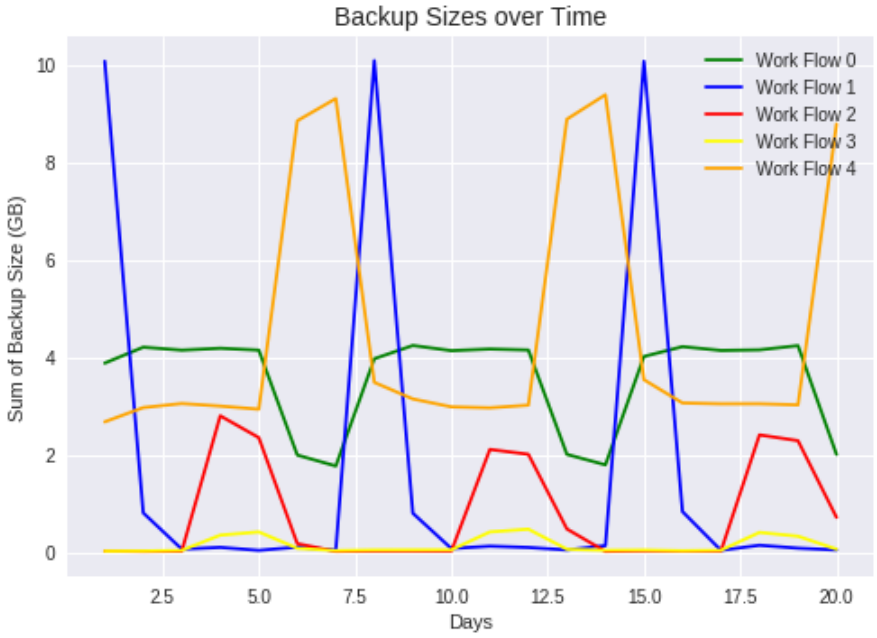
Dataset 1: Network Backup Dataset

---

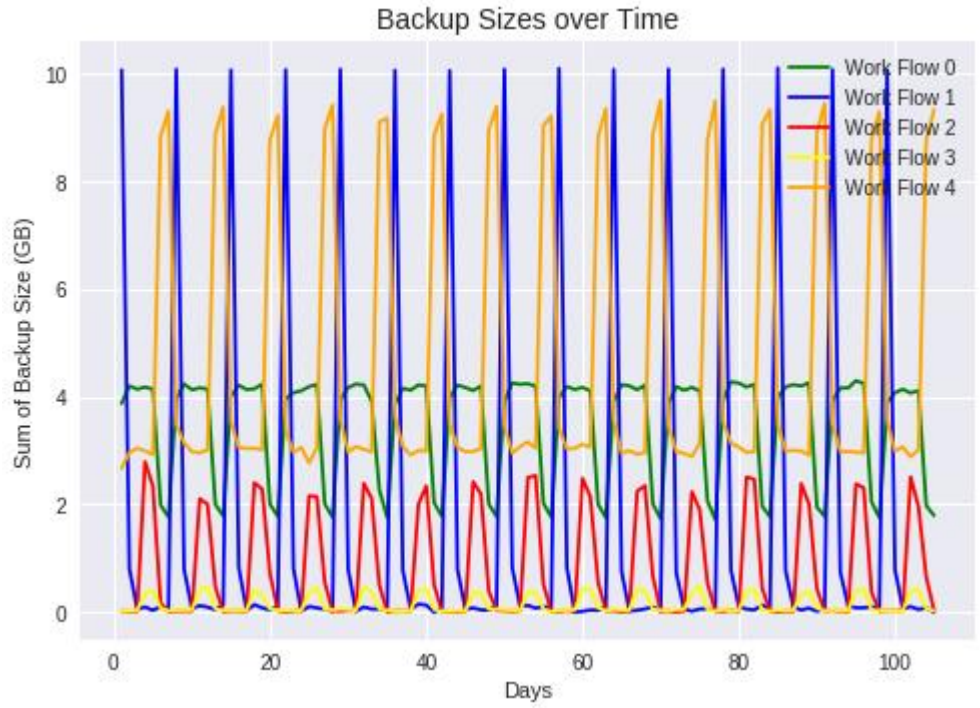
1 Load the Dataset

Network Backup Dataset is comprised of simulated traffic data on a backup system over a network. This system monitors the files and copies their changes every four hours. At the end of each four hours, the system logged the size of the data moved to the destination and the duration it took. We used these data to develop prediction models. We expect the size of backup will have be similar within one workflow because we define a workflow as a task that backs up data from a group of files. The Network Backup Dataset contains 18588 data points with 7 columns. In this project, we aimed to predicting the backup size of the file given the other attributes.

(a) For the first twenty-day period (x-axis unit is day number) plot the backup sizes for all workflows (color coded on the y-axis).



(b) Do the same plot for the first 105-day period.



(c) Can you identify any repeating patterns?

The backup sizes for each workflow seem to repeat each week, suggesting that the day of the week can be an important factor to determine the backup size within each workflow, and that backup data within each workflow have similar patterns of change in term of size over time.

2 Prediction

In this section, we used several prediction models to predict the backup size of a file given other attributes. For the design matrix, X, we included Week Number, Day of Week, Backup Start Time - Hour of Day, Work-flow-ID, File Name as our predictor variables, and Size of Backup (GB) as our dependent variable. We further tried different encoding schemes for each feature and in combination.

We encoded the categorical variables (Day of the Week, Hour of the Day, Work-flow-ID, File Name, and Week Number) in two ways:

- Scalar encoding
- One-hot-encoding

For scalar encoding, we convert categorical variables into one-dimensional numerical values. For instance, we convert day of week variable from Monday, Tuesday, …, Sunday to 1, 2, ... ,7. On the other hand, for one-hot-encoding, we convert categorical variables into N dimensional vectors. That is, if a categorical variable has a total of N categories, we can encode it into N variables such that each variable takes only binary value. For instance, for the Day of Week, Monday could be encoded as [1, 0, 0, 0, 0, 0, 0], and Tuesday could be encoded as [0, 1, 0, 0, 0, 0, 0]

(a) Fit a linear regression model. We use ordinary least square as the penalty function.

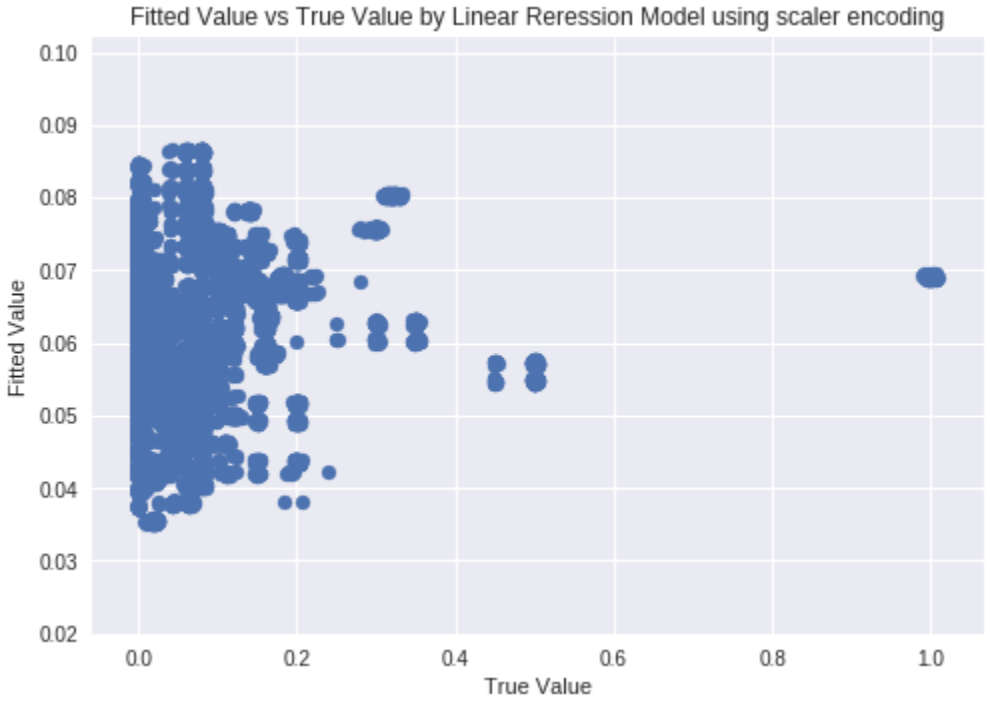
$$\min \|Y - X\beta\|^2$$

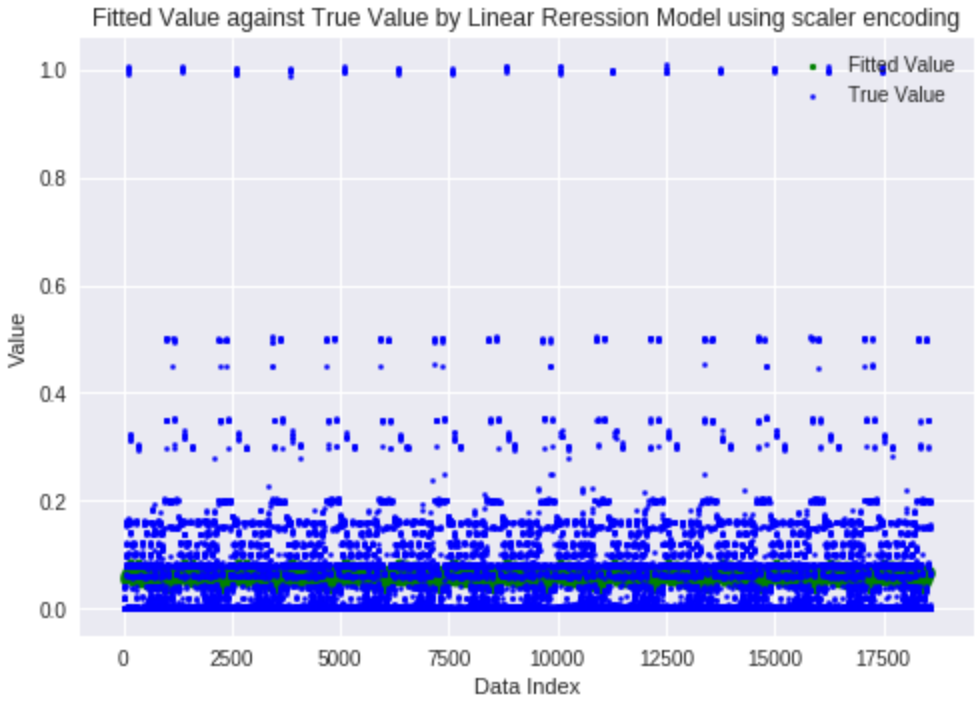
where the minimization is on the coefficient vector β.

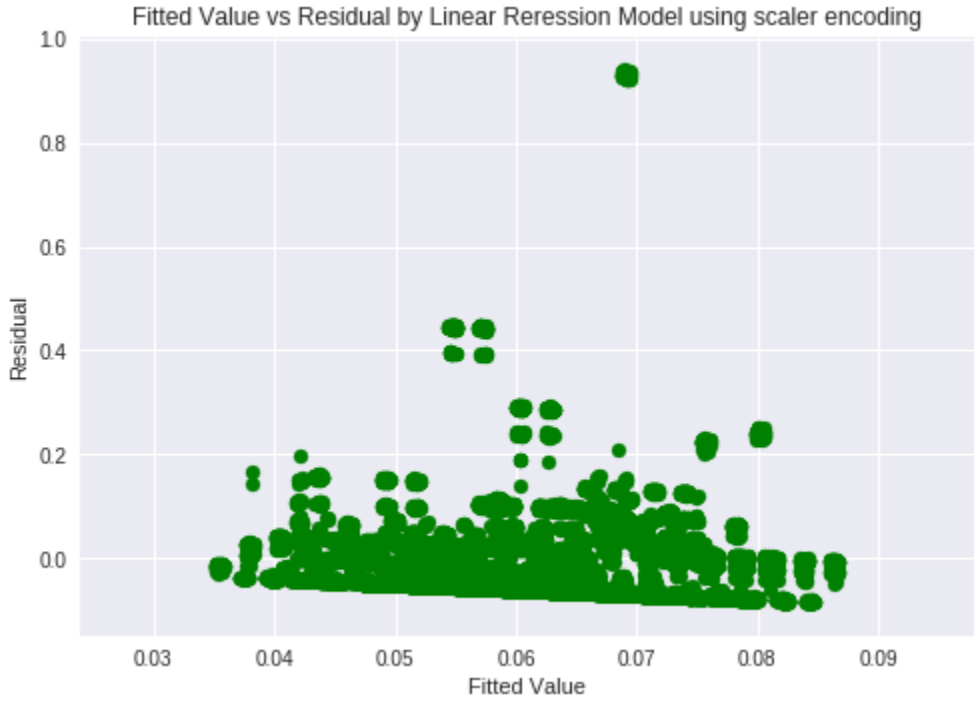
Convert each categorical feature into one dimensional numerical values using scalar encoding (e.g. Monday to Sunday can be mapped to 1-7), and then directly use them to fit a basic linear regression model.

Scatterplots of the fitted values against the true values by linear regression model using scaler encoding was shown below. The model did not fit the data well since most fitted values were greater than true values and they were not close to each other at all. From the second scatterplot, some overlap can be seen when the fitted and true values were in the lower range, from 0.0 to 0.2.

In the residual plot shown as below, most residuals are around 0 but they are not evenly distributed around the 0 line, which indicates non-linearity.







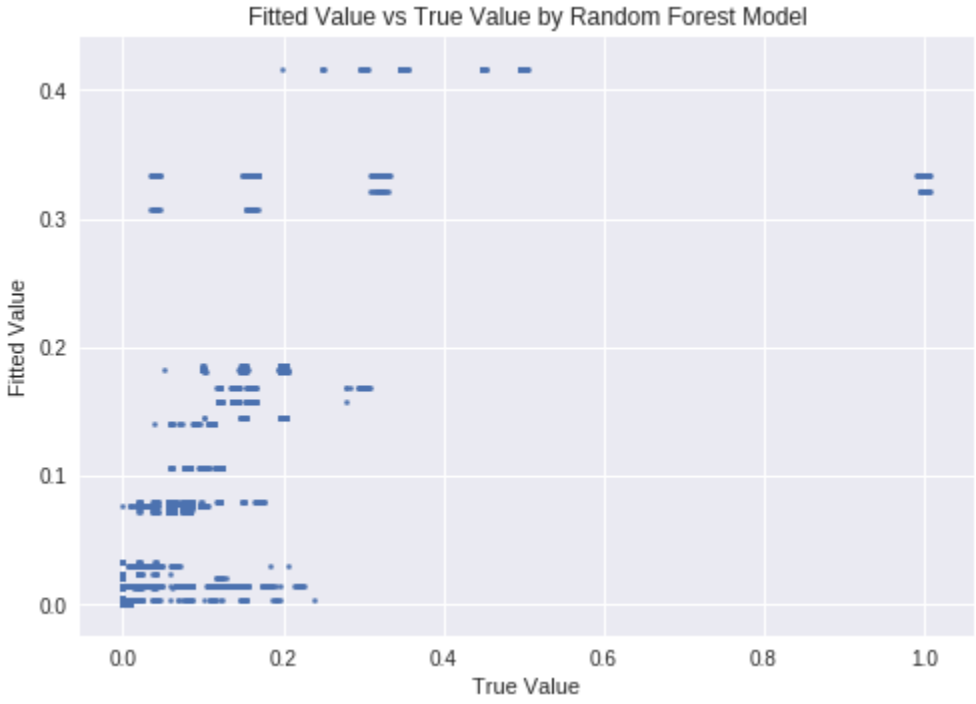
Report training and test Root Mean Squared Error (RMSE) from 10-fold cross validation

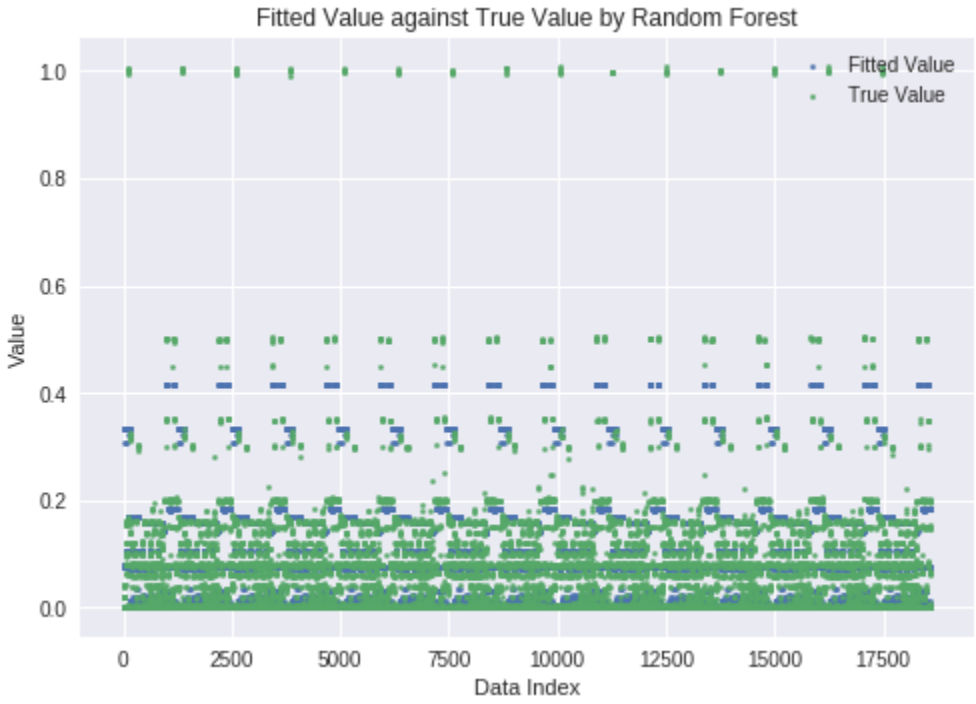
```
Test RMSE [0.10671805 0.10018461 0.10684977 0.10036709 0.10711585 0.10044534
 0.10705027 0.10046665 0.10707419 0.09994712]
Train RMSE [0.10324316 0.10396678 0.1032258  0.10394643 0.10319511 0.10393838
 0.10320263 0.10393639 0.10320099 0.1039916 ]
```

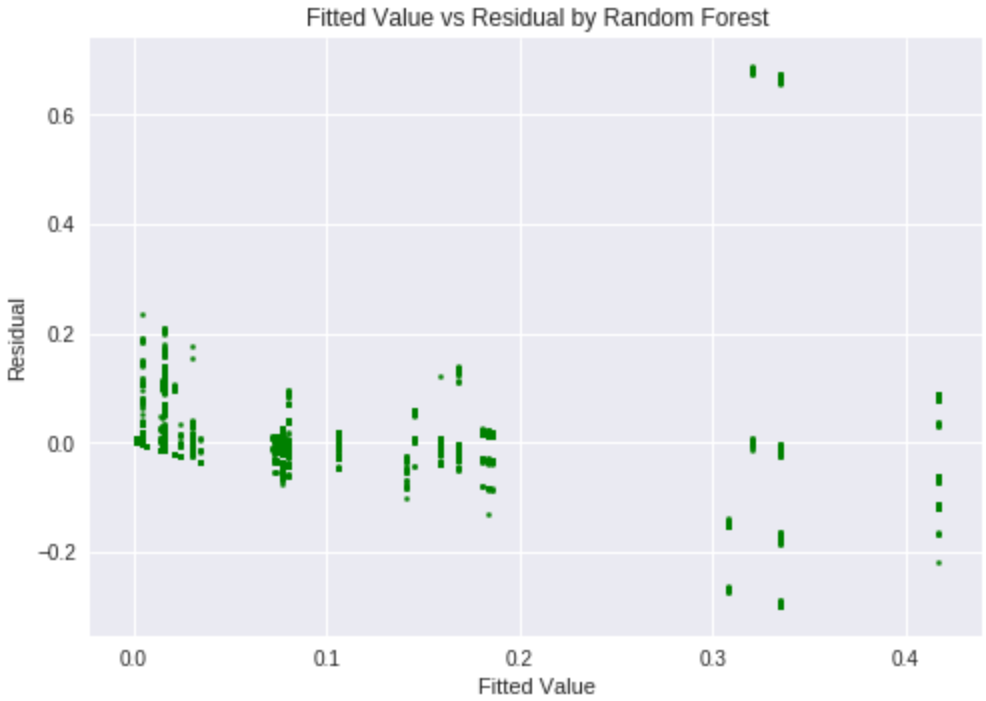
(b) Use a random forest regression model for the same task.

Scatterplots of fitted value against true values by random forest regression model were shown below. The random forest model fit the data better than the linear regression model since more fitted values were closer to the true value in the first scatterplot. In the second scatterplot, more fitted values in the range of (0, 0.6) overlapped with the true values. The range of overlap was bigger than that of the linear model.

In the residual plot, most residuals were around the 0 line and the distribution compared to the linear model was less uneven around the 0 line, which means that the residuals had a better distribution than those in the linear model.







Report training and test Root Mean Squared Error (RMSE) from 10-fold cross validation

```
Test RMSE from 10 fold cross validation [0.06764074 0.05154571 0.06745133 0.053318
0.06652498 0.05391141 0.06774105 0.05267539 0.06740098 0.0533911 ]
Train RMSE from 10 fold cross validation [0.0601408  0.05960861 0.06013765
0.0616376  0.05937561 0.06088864 0.06012154 0.06117908 0.06017252 0.0615678 ]
```

i. Report Training and average Test RMSE from 10 fold cross validation (sum up each fold's square error, divide by total number of data then take square root) and Out Of Bag error you get from this initial model.

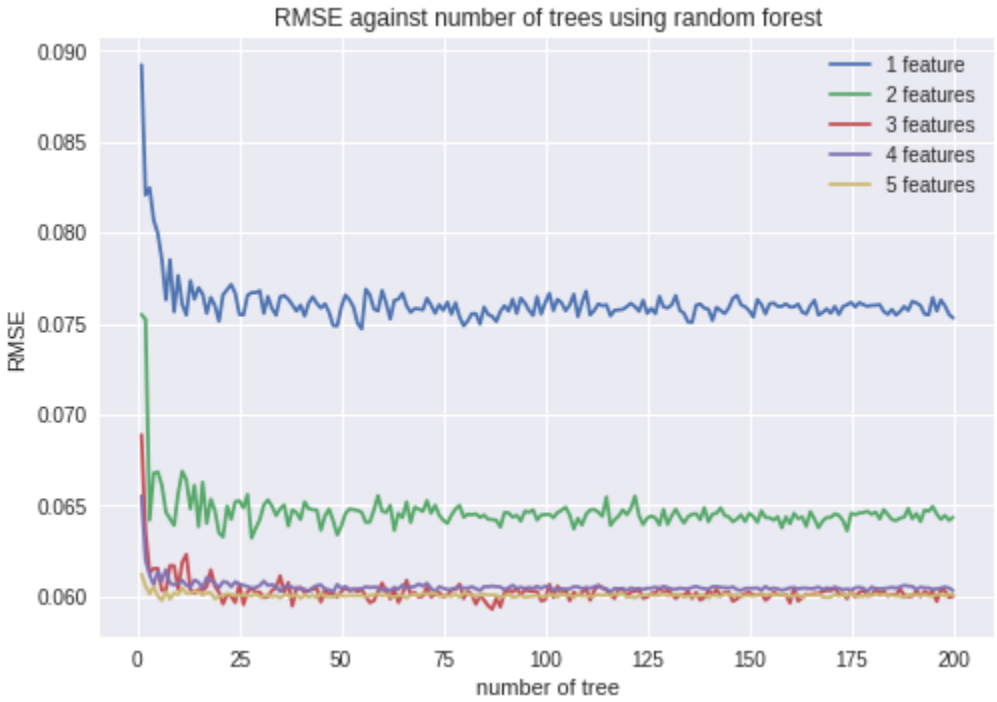
```
Average Test RMSE from 10 fold cross validation 0.0601600701074104
Average Train RMSE from 10 fold cross validation 0.060482985344622486
Out of Bag Error from 10 fold cross validation [0.33661706 0.32972731 0.33712365
0.349956   0.33030091 0.34151442 0.33706991 0.34518871 0.3381595  0.34758671]
Average Out of Bag Error from 10 fold cross validation 0.3393244182507348
Out of Bag Error 0.33840610333779064 from fitting the entire available dataset
```

ii. Sweep over number of trees from 1 to 200 and maximum number of features from 1 to 5, plot

1) out-of-bag error (y axis) against number of trees (x axis)

The two graphs below showed that both out-of-bag error and RMSE decreased drastically as the number of tree goes from 1 to 25, and became stable after number of tree reaches 25. The graphs also showed that 3 features could be an optimal choice to achieve the best performance.

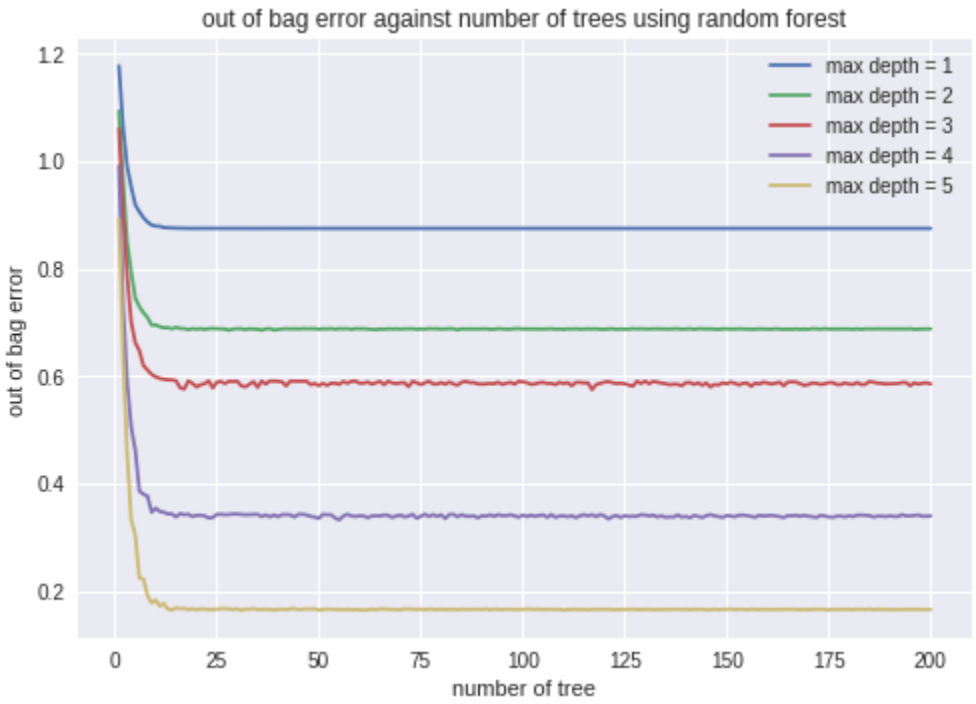
2) average test-RMSE (y axis) against number of trees (x axis)



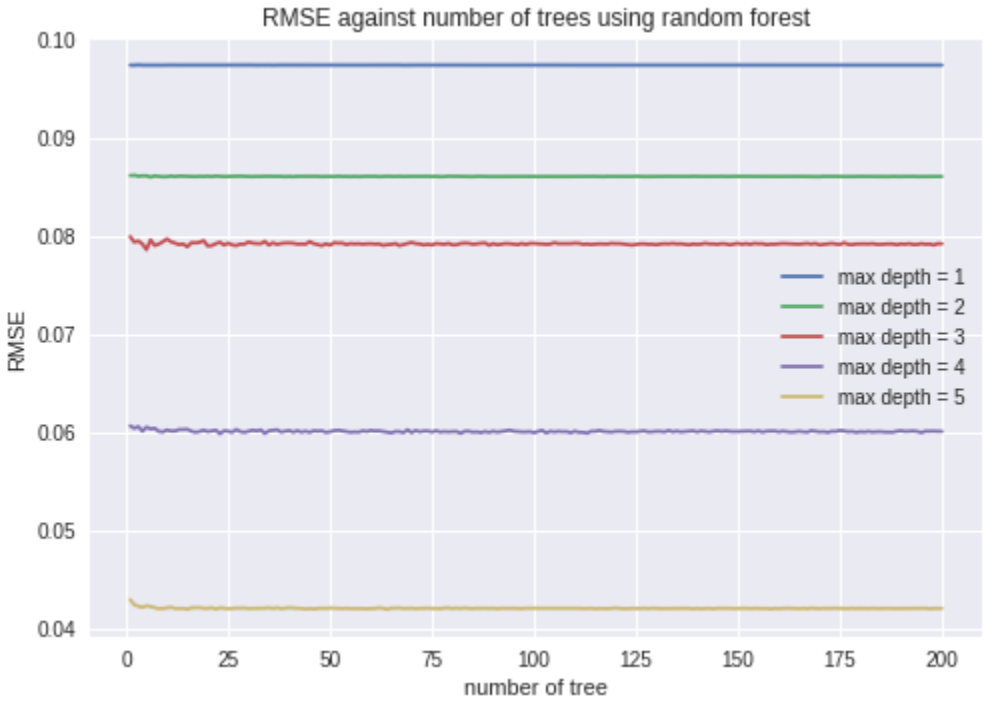
iii. Pick another parameter you want to experiment on. Plot similar figure 1 and figure 2 as above. What parameters would you pick to achieve the best performance?

Here we select maximum depth as the parameter to experiment on. We plotted the out-of-bag error against number of trees and average RMSE against number of trees. Based on the two graphs below, we can see that with maximum depth is 5, the model achieve the best performance. The graphs also show that the out-of-bag errors decrease drastically between number of trees is 1 and 20, and become stable after number of trees reach 20.

1) out-of-bag error (y axis) against number of trees (x axis)



2) average test-RMSE (y axis) against number of trees (x axis)



iv. Report the feature importances you got from the best random forest regression you find.

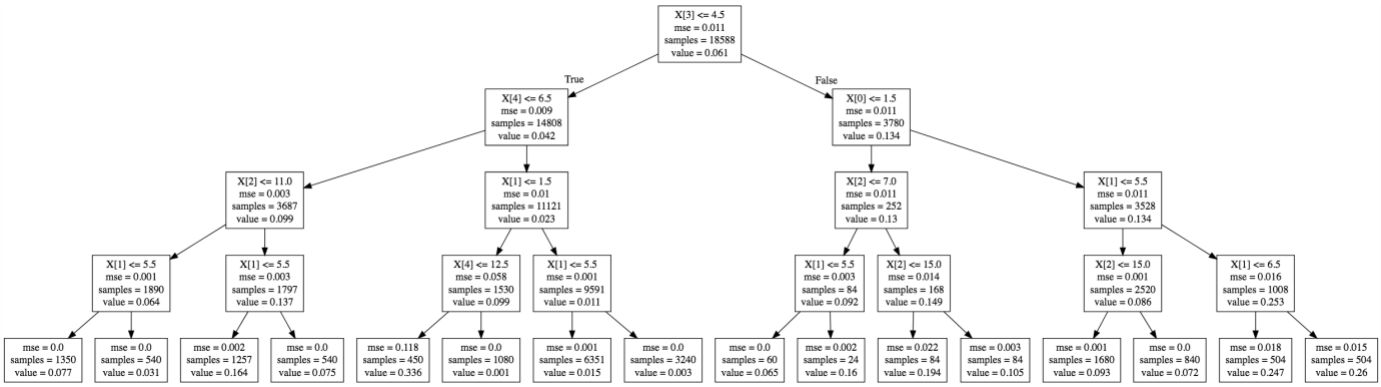
The best combination of the parameters we found from the previous hyperparameter search is with maximum depth = 5, 25 trees, and 3 features. We fitted the model with the scalar-encoding design matrix and derived a table below with feature importance. The average RMSE across 10 fold is `0.0448119411140924`.

Week #	Day of Week	Backup Start Time	Work Flow ID	File Name
0.00078131	0.31868635	0.23028964	0.1799154	0.27032731

Based on the table above, we can rank the importance of the feature by feature importance. The higher the number is, the more important the feature is. We can see that Day of Week is the most important feature, and week number is the least important feature. This intuitively makes sense because the activity of network generally depends on the day of week, and it remain about the same regardless of which week we are in. The observation also matches up with the graphs from part 1 where the patterns of the backup size repeat every week.

v. In our decision tree, we see that the root node is based on the 4th feature, which is not the most important feature from our previous node. However, we do see that most of the nodes in the final level are based on the most important feature, Day of the Week

Scalar encoding



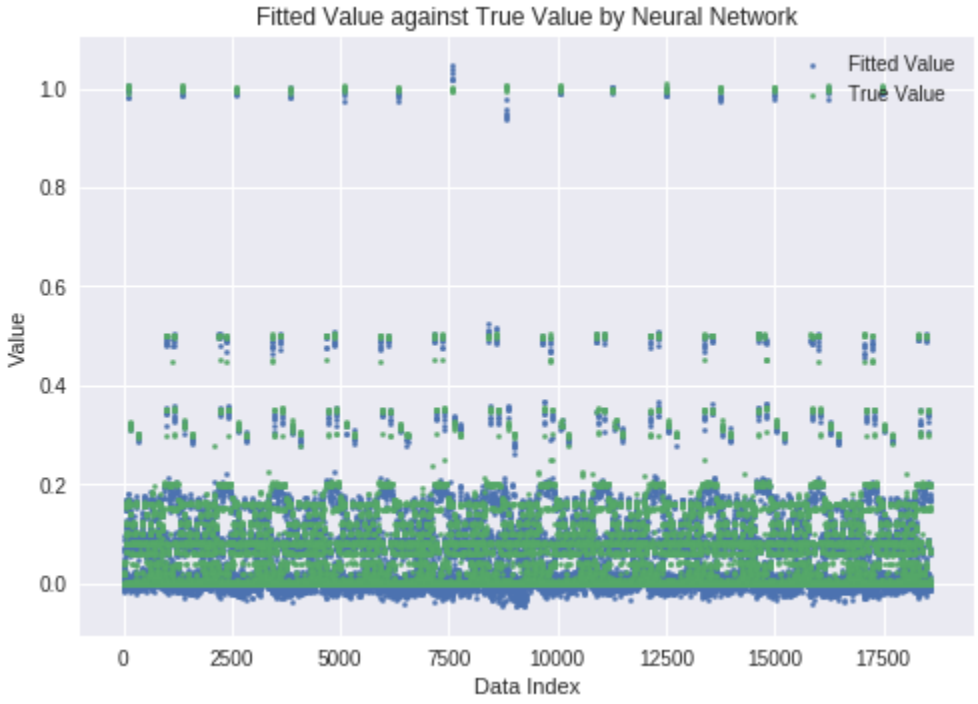
(c) Now use a neural network regression model (one hidden layer) with all features one-hot encoded. Parameters:

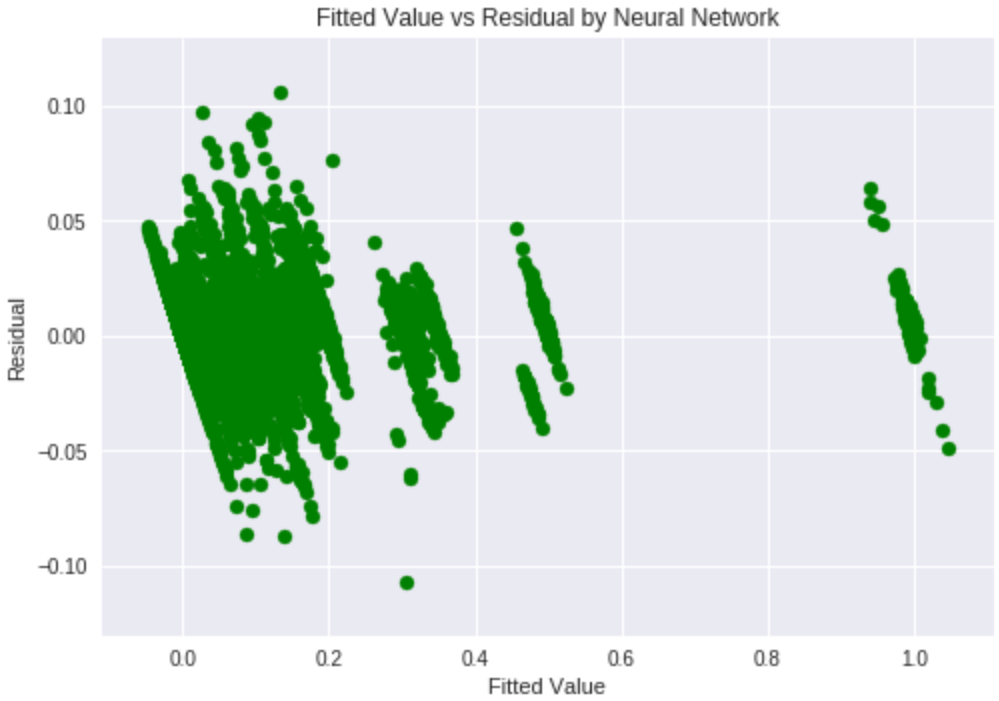
- **Number of hidden units. Try [2, 5, 10, 50, 100, 150, 200, . . . , 600].**
- **Activation function ('relu', 'logistic', 'tanh')**

Scatterplots of fitted value against true values by neural network regression model were shown below. The neural network model fit the data better than the random forest and the linear regression model since almost all of the scatters fell on diagonal in the first scatterplot. In the second scatterplot, more fitted values in the range of (0, 1.2) overlapped with the true values. The range of overlap was bigger than that of the random forest and the linear regression model.

In the residual plot, most residuals were around the 0 line and are evenly distributed around the 0 line, which means that the residuals had a better distribution than those in the random forest and the linear regression model.


Fitted Value vs True Value by Neural Network
Fitted Value
True Value
0.0
0.2
0.4
0.6
0.8
1.0


Fitted Value against True Value by Neural Network
Fitted Value
True Value
Value
Data Index
0
2500
5000
7500
10000
12500
15000
17500



Report training and test Root Mean Squared Error (RMSE) from 10-fold cross validation

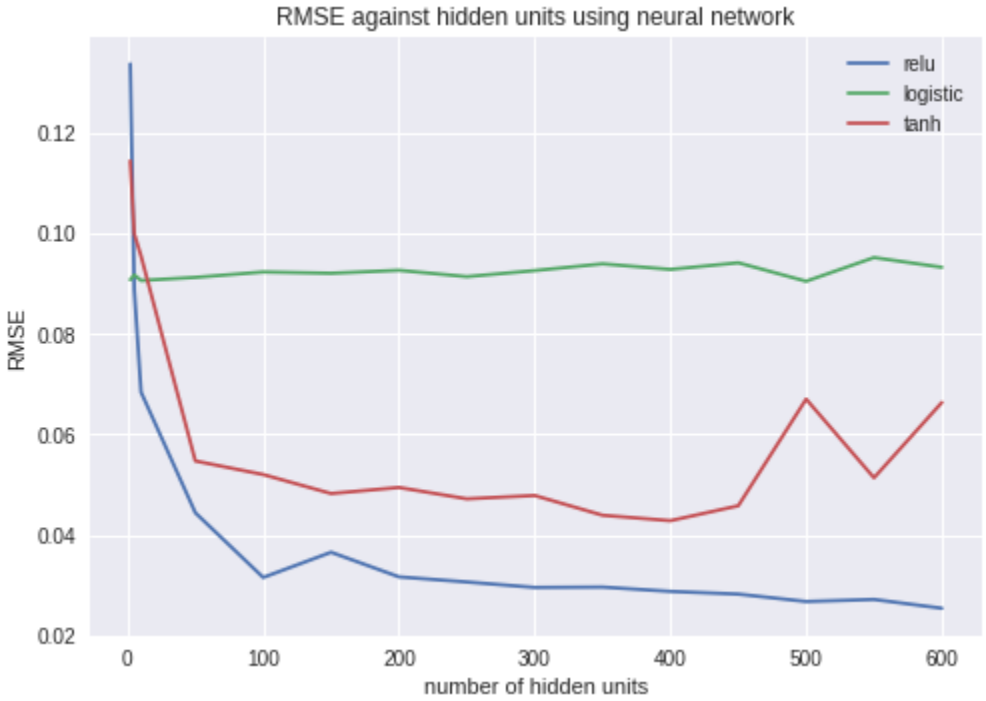
```
Test RMSE
[[0.13535922 0.08157744 0.07011585 0.03833828 0.03144907 0.03554009
  0.02630444 0.03126395 0.02767673 0.02726537 0.02858979 0.0264829
  0.02789353 0.02423361 0.02640424]
 [0.09127432 0.09118132 0.09082181 0.09187211 0.09288199 0.09062986
  0.09173896 0.09369287 0.09148535 0.0932247  0.0929113  0.09160086
  0.09274727 0.09410862 0.09421681]
 [0.10908557 0.10565868 0.10023602 0.05808106 0.0537363  0.04883209
  0.05265533 0.05550875 0.05150909 0.04305396 0.05573796 0.05363166
  0.05705181 0.04594222 0.05779803]]

Train RMSE
[[0.09630325 0.05662238 0.04486227 0.01849351 0.01649988 0.0157288
  0.01453461 0.01391836 0.01306458 0.0130785  0.01259659 0.01204169
  0.0120707  0.01221741 0.01137639]
 [0.08954402 0.08826114 0.08826893 0.0884945  0.08880116 0.08889117
  0.08927792 0.09096335 0.08956466 0.08956887 0.09051924 0.09007662
  0.09082437 0.09130804 0.09246136]
 [0.08860247 0.08756237 0.08197808 0.03881885 0.03763773 0.03167646
  0.02884183 0.03489568 0.03132709 0.02775407 0.04027087 0.0334895
  0.04118098 0.03694584 0.04811161]]
```

Plot test-RMSE vs the number of hidden units for each activation function. Report the best combination.

The plot of the test-RMSE vs the number of hidden units using neural network for each activation function was shown below. For the logistic activation function, the test-RMSE barely changed, which was around 0.09. For the relu and tanh activation functions, the test-RMSE decreased as the number of

hidden units increased. In general, the test-RMSE of relu activation function was smaller than those of logistic and tanh. Based on this plot, the best combination is to use relu as the activation function when the number of hidden units is 600 since it reached the lowest RMSE.



(d) Predict the Backup size for each of the workflows separately.

i. Using linear regression model. Explain if the fit is improved?

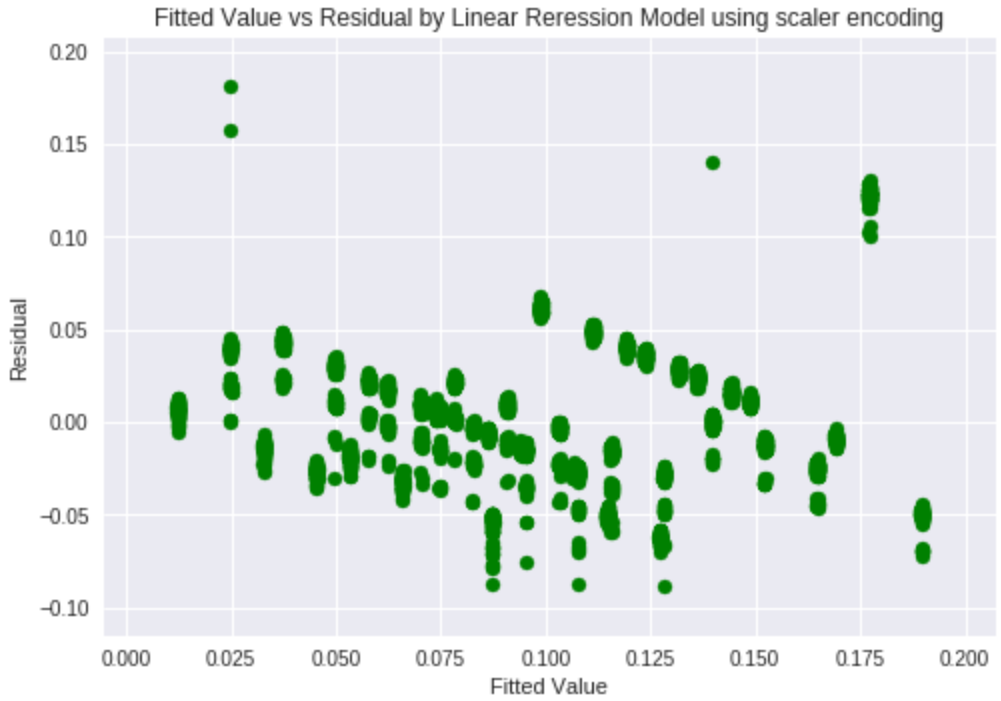
In general, the improvement of the fit varied a lot for different workflows. The fit improved the most in workflow ID = 0 and didn't improve much in workflow ID = 2, 3. This suggests that the different workflows are not linearly related. This probably explains why the simple linear regression didn't fit the data well. A more complex regression model may be necessary in order to fit the data better.

```
Work Flow ID =  0
training RMSE =  [0.03570555 0.03594719 0.03572458 0.03570646 0.03577678
0.03602217 0.03566539 0.03602294 0.03575094 0.03603034]
test RMSE =  [0.03705092 0.03486967 0.03687099 0.0370181  0.03640268 0.03417222
 0.03737511 0.03415557 0.03663592 0.0340835 ]
average test RMSE =  0.03586346655296642
```

The fit improved the most in workflow ID = 0. Most values in the first scatterplot fell on the diagonal. In the second scatterplot most fitted values and true values overlapped. In the residual plot the residuals distributed evenly around line 0.


Fitted Value vs True Value by Linear Reression Model using scaler encoding
Fitted Value
True Value
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.00
0.05
0.10
0.15
0.20
0.25
0.30

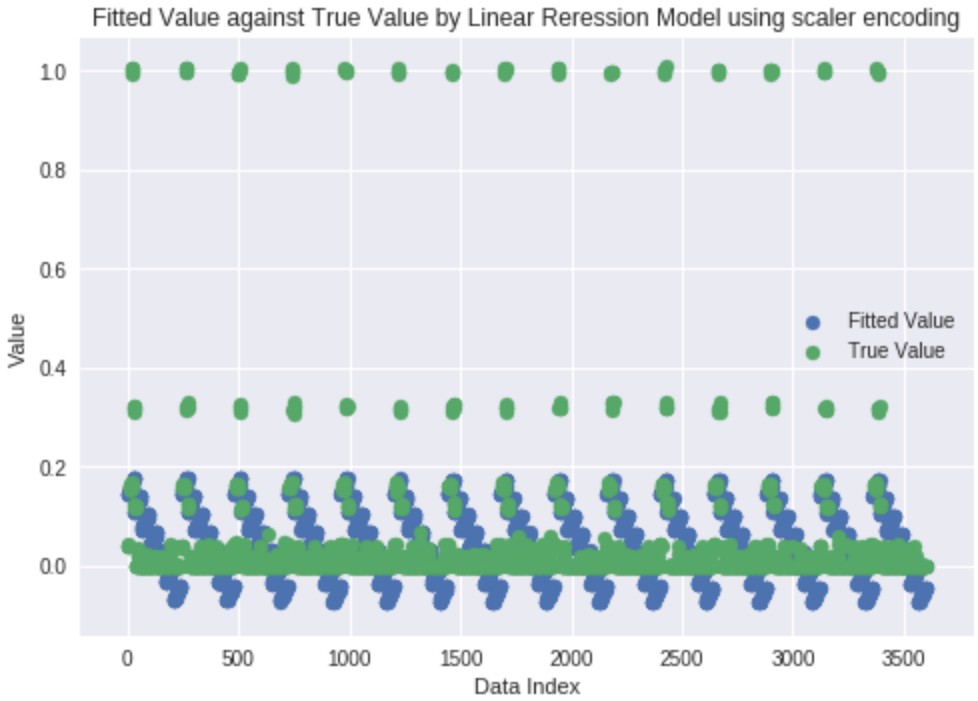

Fitted Value against True Value by Linear Reression Model using scaler encoding
Fitted Value
True Value
Value
Data Index
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
500
1000
1500
2000
2500
3000
3500

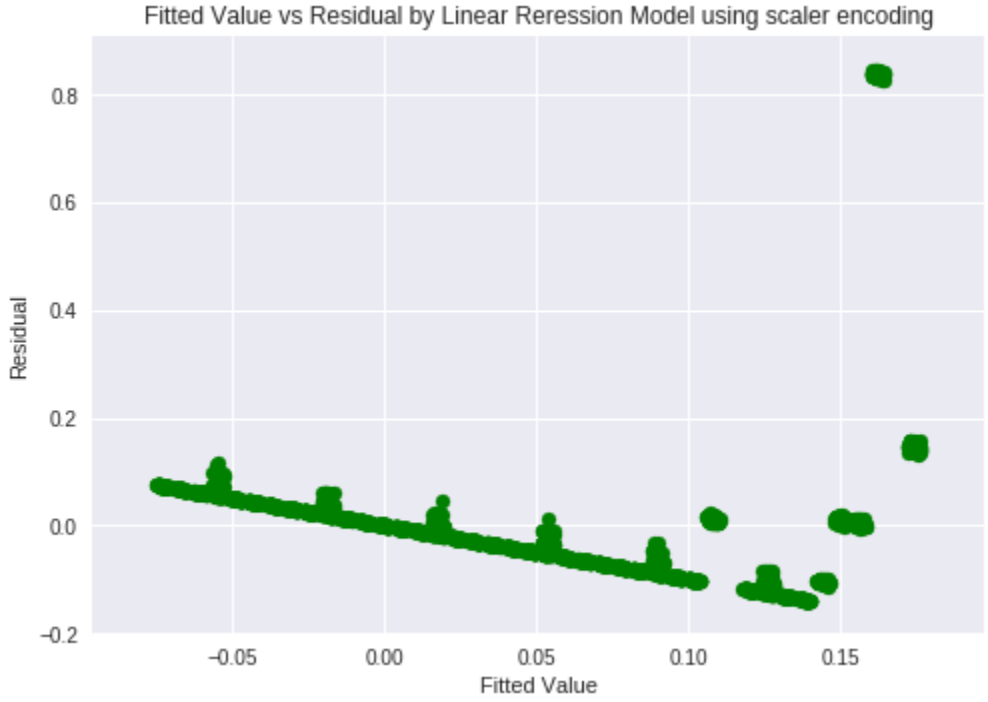


Work Flow ID = 1

```
training RMSE =  [0.1461789  0.15130557 0.14620991 0.1513097  0.14621229
0.15126452 0.14618282 0.15130344 0.14617777 0.15129651]
test RMSE =  [0.17043391 0.12379992 0.17017383 0.12371056 0.17014842 0.1242011
 0.17038135 0.12378352 0.17043517 0.12390809]
average test RMSE =  0.14709758822040112
```

The fit in workflow ID = 1 improved more than ID = 2, 3 but not as much as the improvement in workflow ID = 0, 4. In the residual plot the residuals didn't distribute evenly around 0.

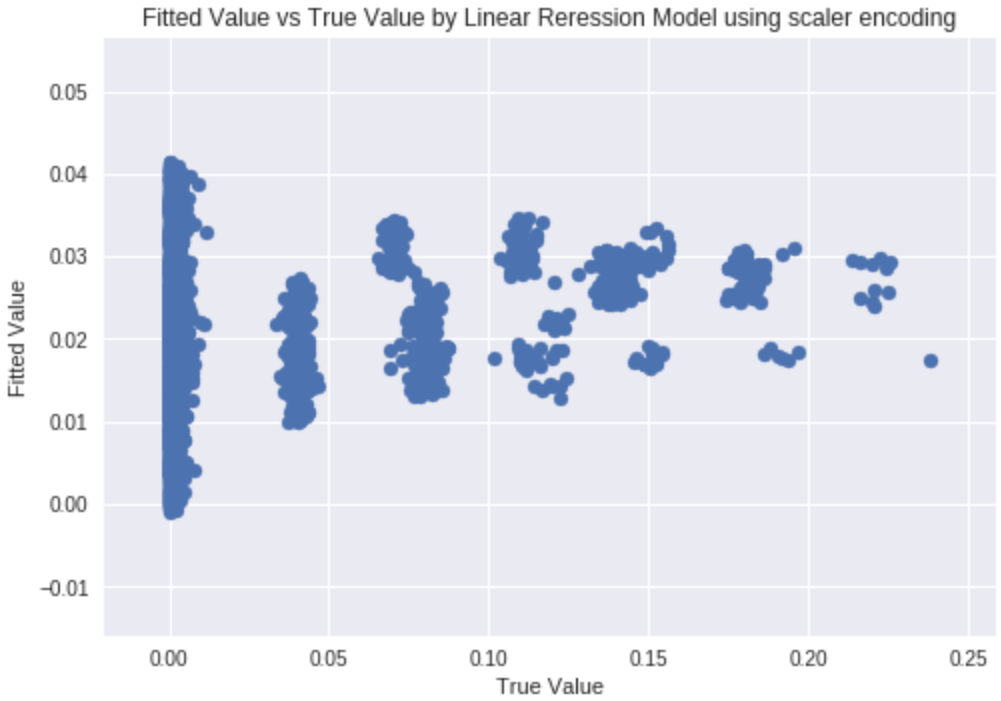


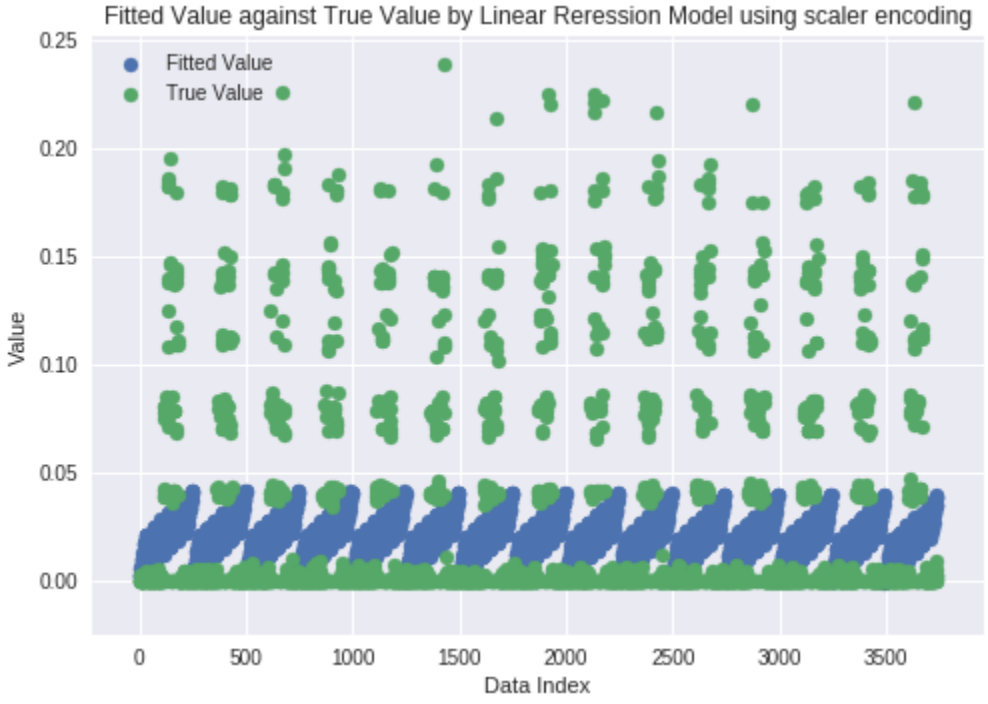


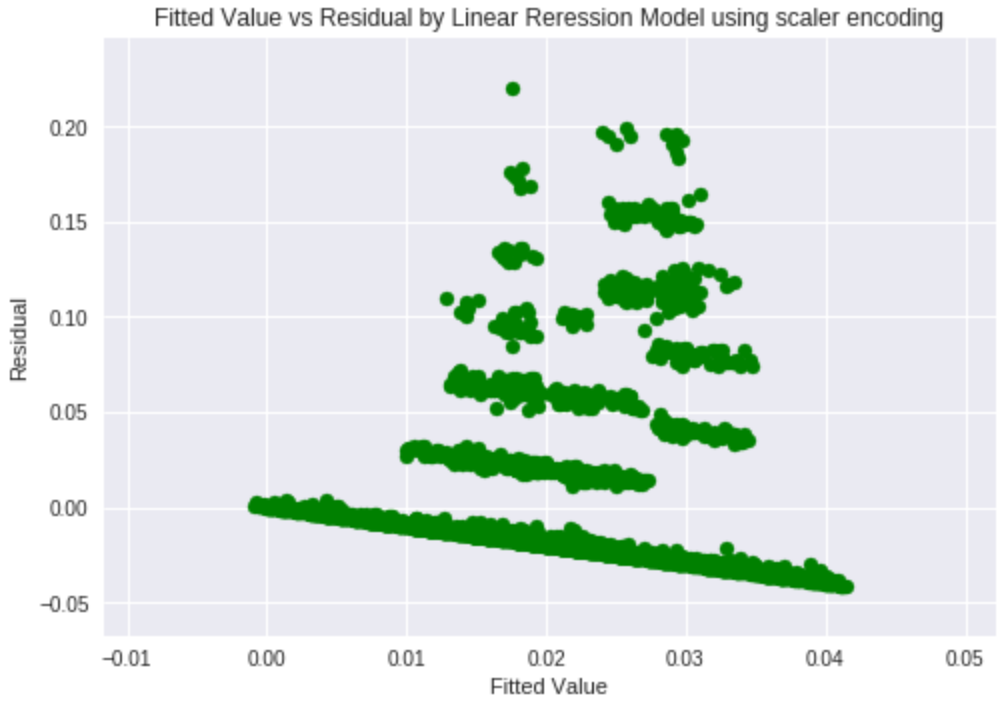
Work Flow ID = 2

```
training RMSE =  [0.04358677 0.04227512 0.04366026 0.04249173 0.04365586
0.04168399 0.04340159 0.04236531 0.04368263 0.04222868]
test RMSE =  [0.0364608  0.0483814  0.03560204 0.04659827 0.03563556 0.05279302
 0.03835048 0.04764129 0.03548782 0.04880264]
average test RMSE =  0.04257533084504969
```

The fit did not improve much in workflow ID = 2. There was little overlap in the second scatterplot among fitted values and true values and the residual plot had a poor distribution as well.


Fitted Value vs True Value by Linear Reression Model using scaler encoding
Fitted Value
True Value
0.05
0.04
0.03
0.02
0.01
0.00
−0.01
0.00
0.05
0.10
0.15
0.20
0.25

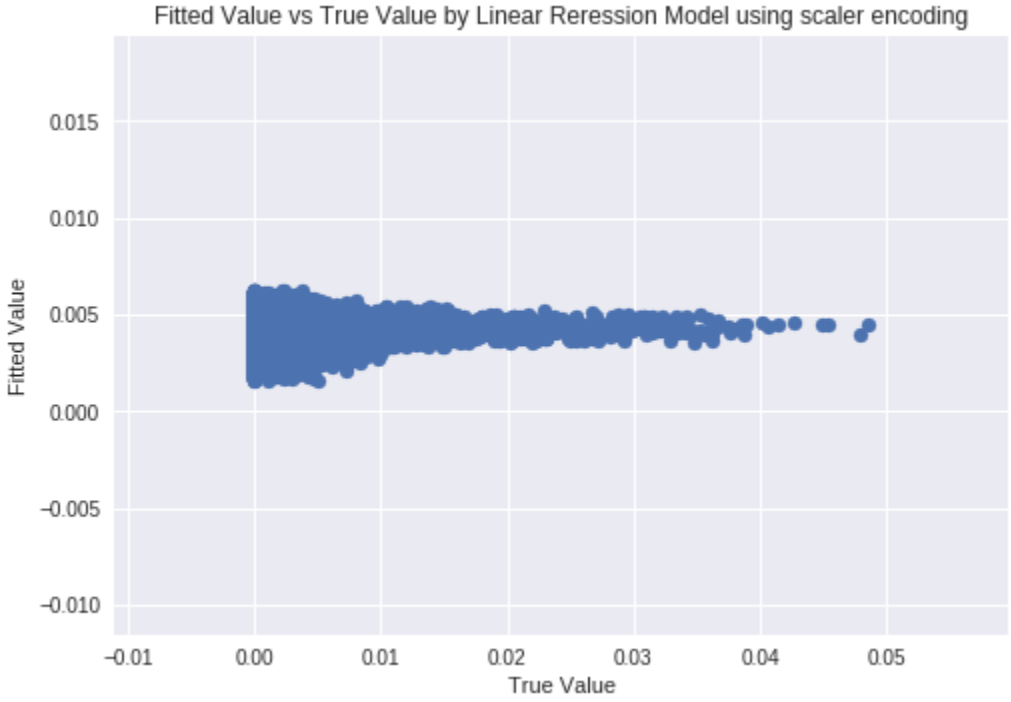

Fitted Value against True Value by Linear Reression Model using scaler encoding
Fitted Value
True Value
Value
Data Index
0.25
0.20
0.15
0.10
0.05
0.00
0
500
1000
1500
2000
2500
3000
3500

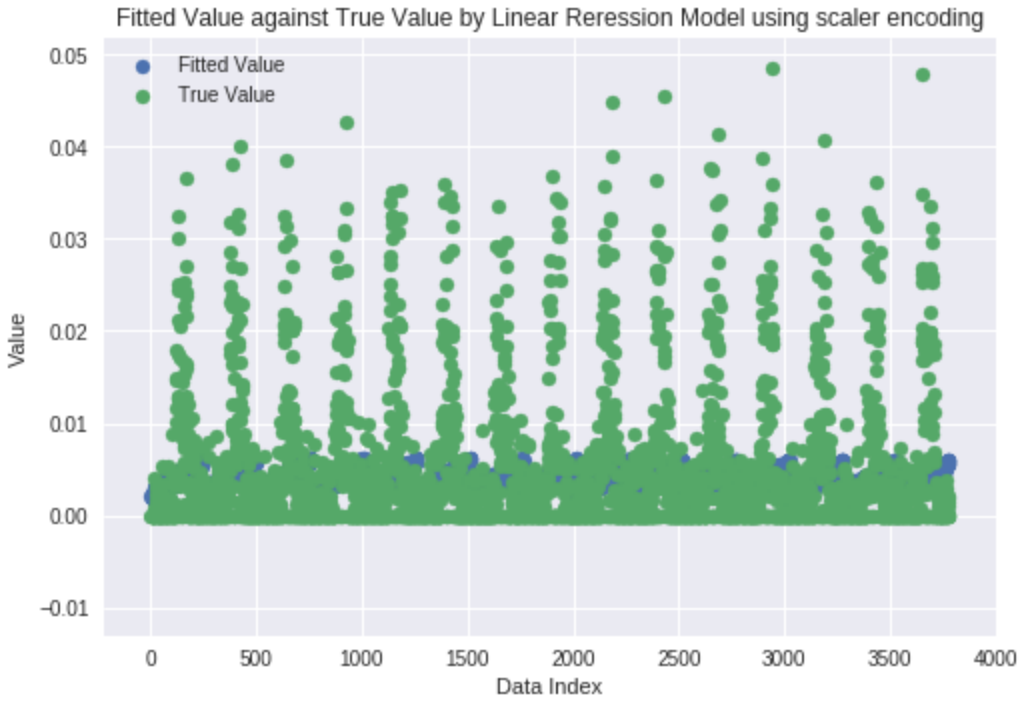


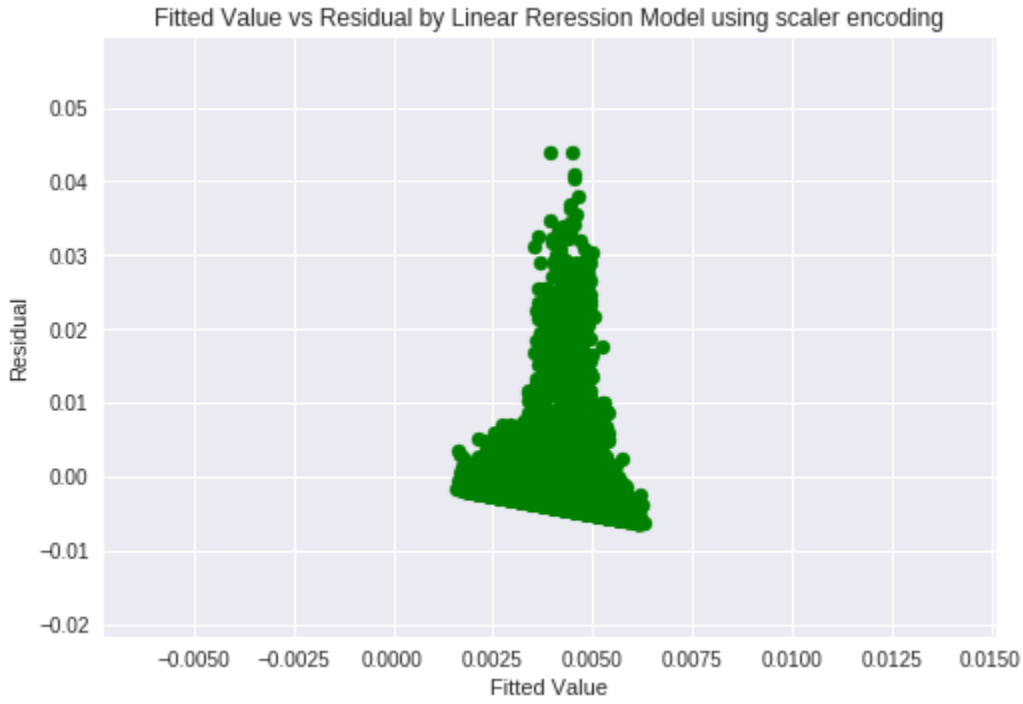
Work Flow ID = 3

```
training RMSE =  [0.00736828 0.00717788 0.00733687 0.00713597 0.00739935
0.00711621 0.00731219 0.00708244 0.00734669 0.00715395]
test RMSE =  [0.00604342 0.00783062 0.0063608  0.00816303 0.00567624 0.00831707
 0.00660988 0.00857479 0.00626729 0.0080237 ]
average test RMSE =  0.007186683488524773
```

The fit did not improve much in workflow ID = 3. There was some overlap in the second scatterplot among fitted values and true values but the residual plot had a poor distribution as well.







Work Flow ID = 4

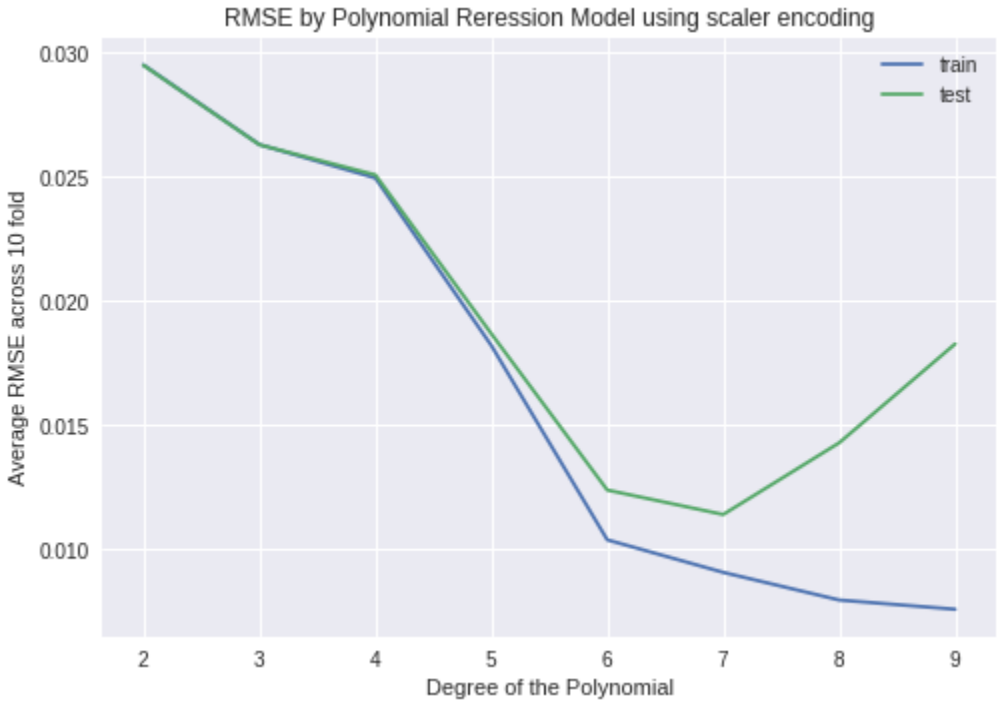
```
training RMSE =  [0.0871622  0.0847293  0.08710513 0.08459278 0.08701006 0.0849066
 0.08704317 0.08475621 0.08706553 0.08476929]
test RMSE =  [0.07390365 0.09604574 0.0745144  0.097119   0.07550607 0.094634
 0.0751583  0.09584044 0.07492722 0.09574873]
average test RMSE =  0.08533975477118065
```

The fit in workflow ID = 4 improved more than ID = 2, 3 but not as much as the improvement in workflow ID = 0. In the residual plot the residuals had some extreme values when the fitted values were high. This

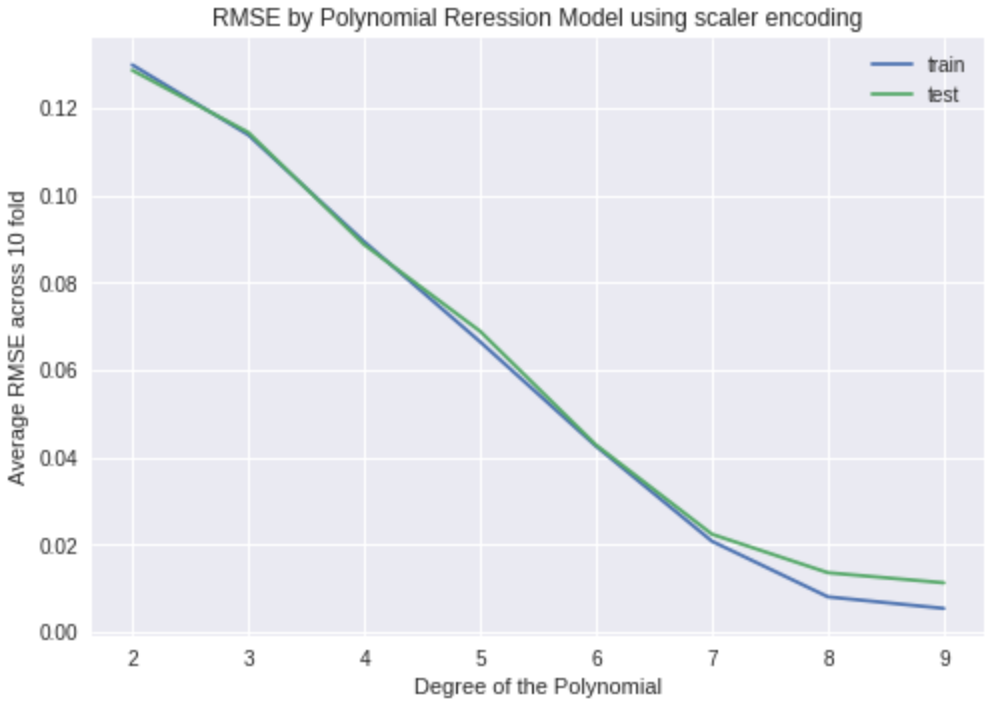
was obvious in the second scatterplot as well, since fitted values were closer to the true values in the lower range and the prediction performed poorly when the true values were high.

ii. The generalization of the model generally gets worse when the degree is increased past 5-7. Cross validation helps to control the complexity model by helping see how the chosen model will generalize to new inputs. If the model generalizes poorly during cross validation, the model may be too complex for the data and features.

```
Work Flow ID =  0
```

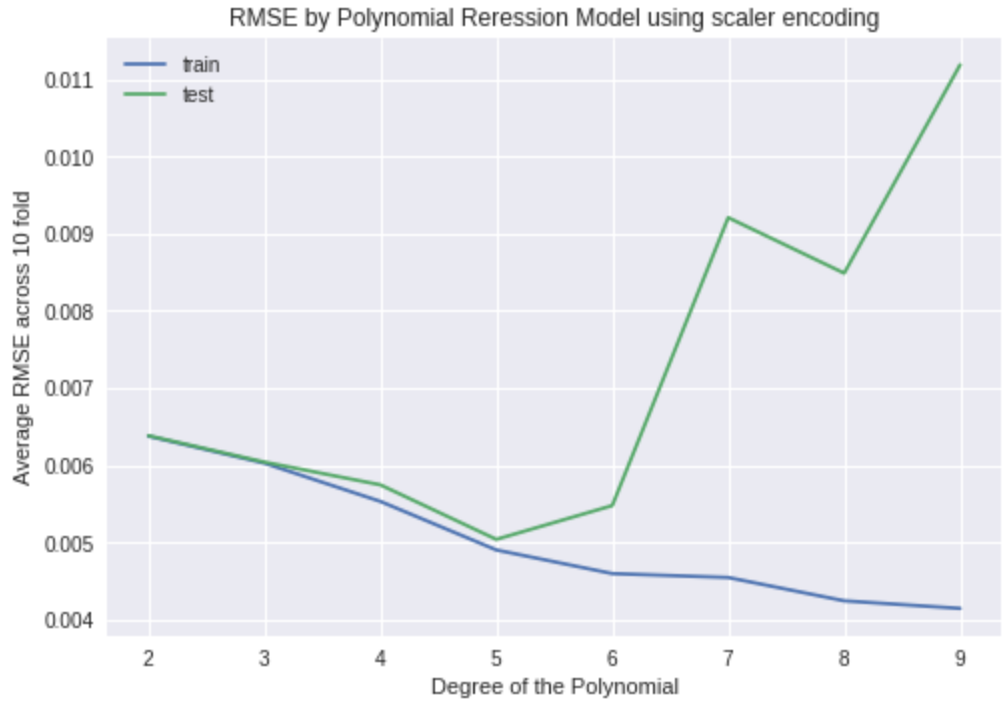



```
Work Flow ID =  1
```

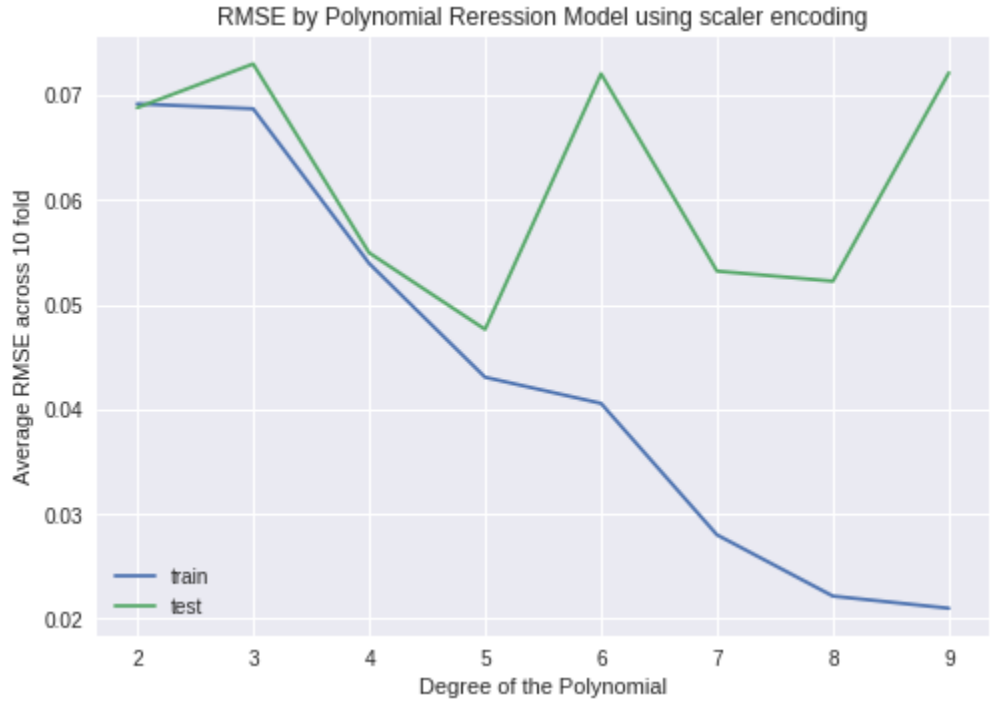


Work Flow ID = 2

Work Flow ID = 3



Work Flow ID = 4



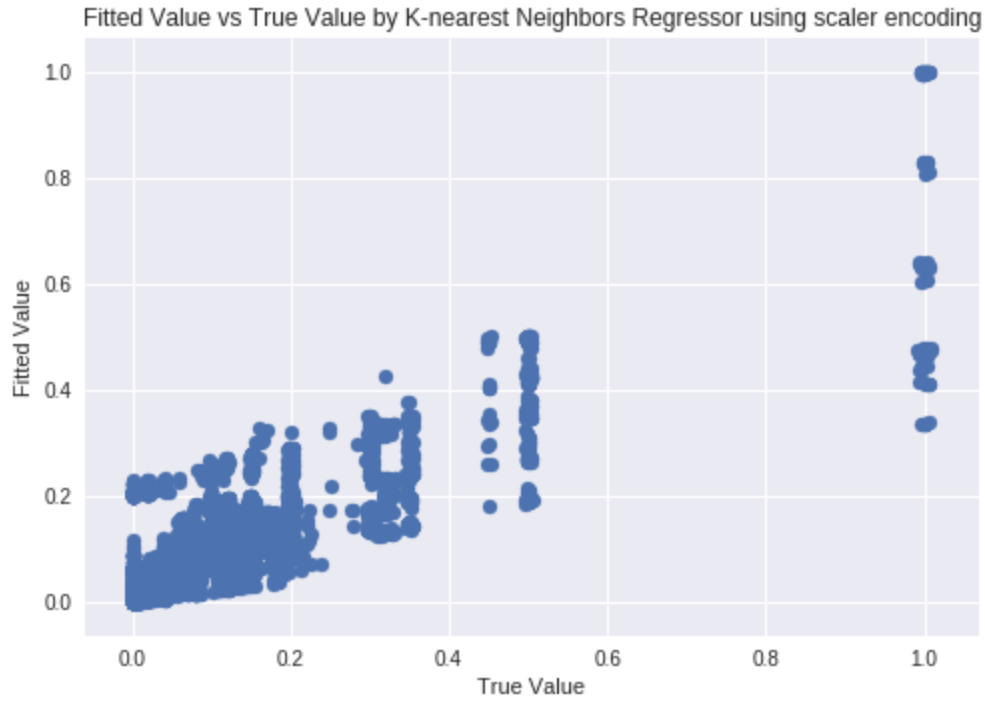
(e) Use k-nearest neighbor regression and find the best parameter.

min test RMSE = 0.014421062366404225

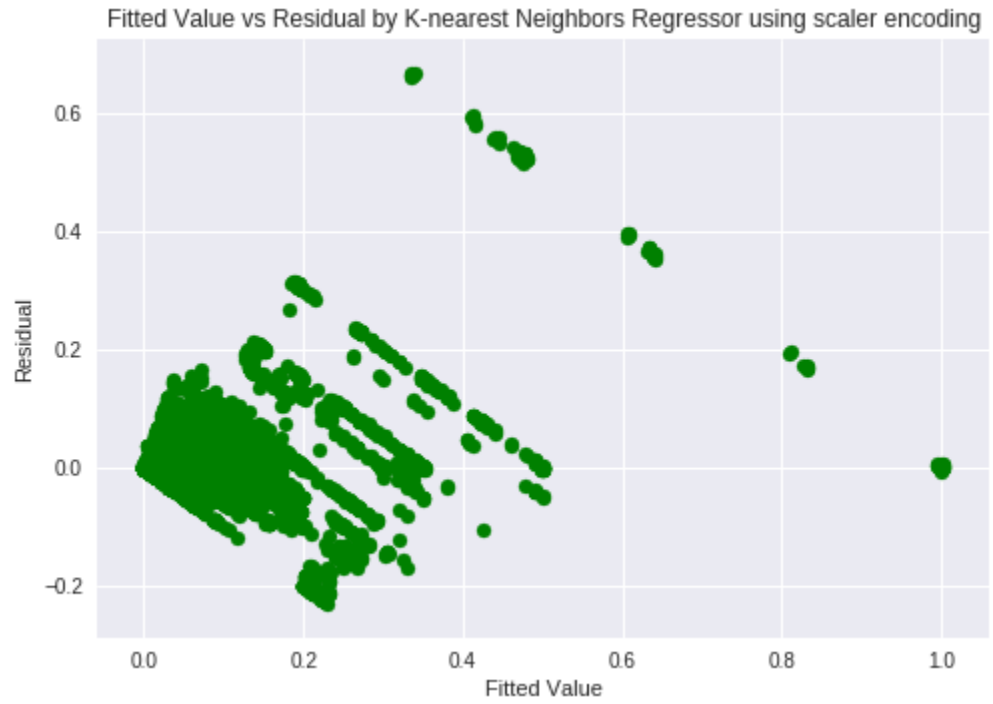
best numbers of neighbors 3

Using KNN regressor with 3 neighbors, we got the RMSE = 0.04799732827665599

Plot fitted values against true values as scatter plots



Plot residuals versus fitted values as scatter plots



Report training and test Root Mean Squared Error (RMSE) from 10-fold cross validation

```
training RMSE =  [0.04802348 0.0493244  0.05090832 0.04951839 0.049126   0.0496706
 0.04746977 0.04880223 0.04899268 0.05036364]
test RMSE =  [0.05204473 0.04163145 0.04804277 0.04227428 0.05021003 0.03153024
 0.04866849 0.04175481 0.04655687 0.01442106]
average test RMSE =  0.04171347292456655
```

3 Comparison

The neural network model seemed to perform the best. This is likely because neural networks are able to handle categorical data and learn highly nonlinear functions. The KNN and random forest models also seemed to do well, but not as well as the neural network. This is likely due to the fact the KNN models are good at handling sparse features, and random forests are good at handling categorical features.

Dataset 2: Boston Housing Dataset

The `Boston Housing Dataset` contains information about housing in the suburbs of the greater Boston area. The dataset contains 506 data points with the following 14 features:

- CRIM: per capita crime rate by town
- ZN: proportion of residential land zoned for lots over 25,000 sq. ft.
- INDUS: proportion of non-retail business acres per town
- CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX: nitric oxides concentration (parts per 10 million)
- RM: average number of rooms per dwelling
- AGE: proportion of owner-occupied units built prior to 1940
- DIS: weighted distances to ve Boston employment centers
- RAD: index of accessibility to radial highways
- TAX: full-value property-tax rate per $10,000
- PTRATIO: pupil-teacher ratio by town
- B: $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town
- LSTAT: % lower status of the population
- MEDV (target variable): Median value of owner-occupied homes in $1000's

1 Load the dataset

The first 5 rows of the dataset are displayed in Table 2.1:

Table 2.1

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2

2 Fit a linear regression model:

(a) Set MEDV as the target variable and the other attributes as the features and ordinary least square as the penalty function.

We fitted a linear regression model using the entire dataset, using MEDV (median home value in $1000s) as the target variable and the remaining 13 attributes as features. The estimated coefficients are displayed in Table 2.2. From the table, we observe that nitric oxide concentration, whether the house is

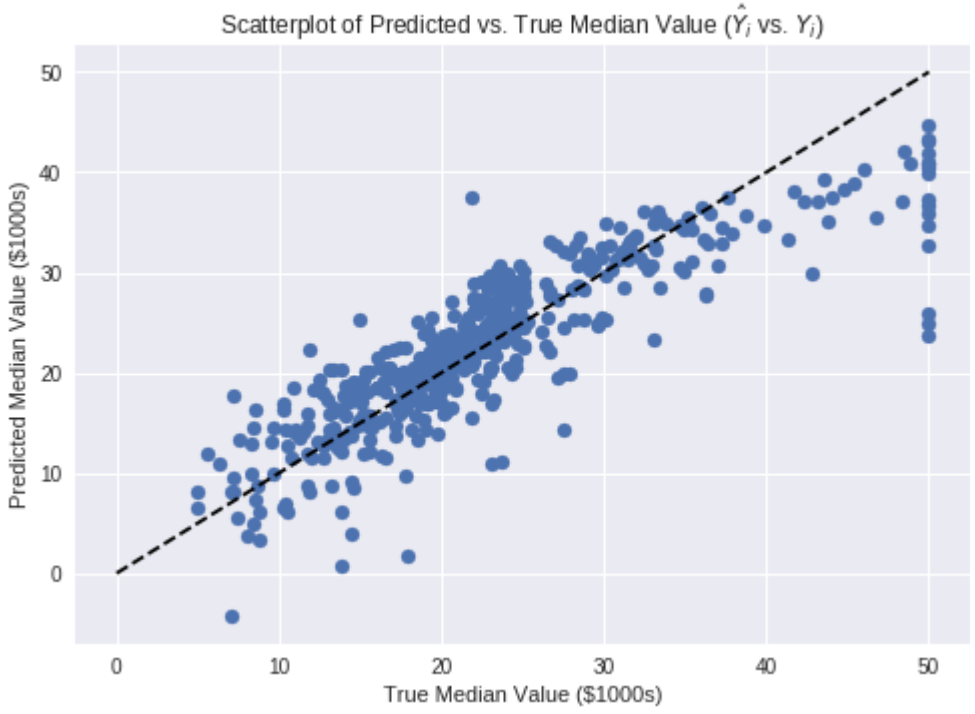
near the Charles river, and number of rooms appear to be features that most strongly affect the median home value (target variable).

Table 2.2

	Features	estimatedCoefficients
0	CRIM	-0.108011
1	ZN	0.046420
2	INDUS	0.020559
3	CHAS	2.686734
4	NOX	-17.766611
5	RM	3.809865
6	AGE	0.000692
7	DIS	-1.475567
8	RAD	0.306049
9	TAX	-0.012335
10	PTRATIO	-0.952747
11	B	0.009312
12	LSTAT	-0.524758

A scatterplot of the fitted values against the true values of the median home value is shown below. We observe that the data appear to be pretty linear and that the fitted and true values are relatively similar, although the model appears to underestimate the higher true median home values.

Figure 2.1



(b) Perform a 10 fold cross validation, analyze the significance of different variables with the statistics obtained from the model you have trained, and the averaged Root Mean Squared Error (RMSE), and plot

1) fitted values against true values as scatter plots using the whole dataset;

2) residuals versus fitted values as scatter plots using the whole dataset.

The table below shows the estimated coefficients, standard errors, t-statistics, p-values, and 95% confidence intervals of the coefficient estimates from the best model obtained via 10-fold cross-validation. The results indicate that all of the variables are highly significant at alpha=0.05 significance level, except INDUS (the proportion of non-retail business acres per town) and AGE (proportion of owner-occupied units built prior to 1940). We see that CRIM, NOX, DIS, TAX, PTRATIO, and LSTAT are associated with a significant decrease in the median home value. ZN, CHAS, RM, RAD, and B are associated with a significant increase in the median home value.

Table 2.3

	coef	std err	t	P>|t|	[0.025	0.975]
const	35.5666	5.417	6.565	0.000	24.920	46.214
CRIM	-0.1056	0.035	-3.005	0.003	-0.175	-0.037
ZN	0.0491	0.014	3.432	0.001	0.021	0.077
INDUS	0.0322	0.066	0.486	0.627	-0.098	0.162
CHAS	2.5129	0.894	2.812	0.005	0.756	4.269
NOX	-17.6280	4.233	-4.165	0.000	-25.947	-9.309
RM	3.8177	0.439	8.702	0.000	2.955	4.680
AGE	0.0106	0.014	0.745	0.457	-0.017	0.038
DIS	-1.4361	0.211	-6.805	0.000	-1.851	-1.021
RAD	0.3615	0.083	4.368	0.000	0.199	0.524
TAX	-0.0155	0.005	-3.431	0.001	-0.024	-0.007
PTRATIO	-0.9123	0.141	-6.483	0.000	-1.189	-0.636
B	0.0099	0.003	3.295	0.001	0.004	0.016
LSTAT	-0.5551	0.054	-10.295	0.000	-0.661	-0.449

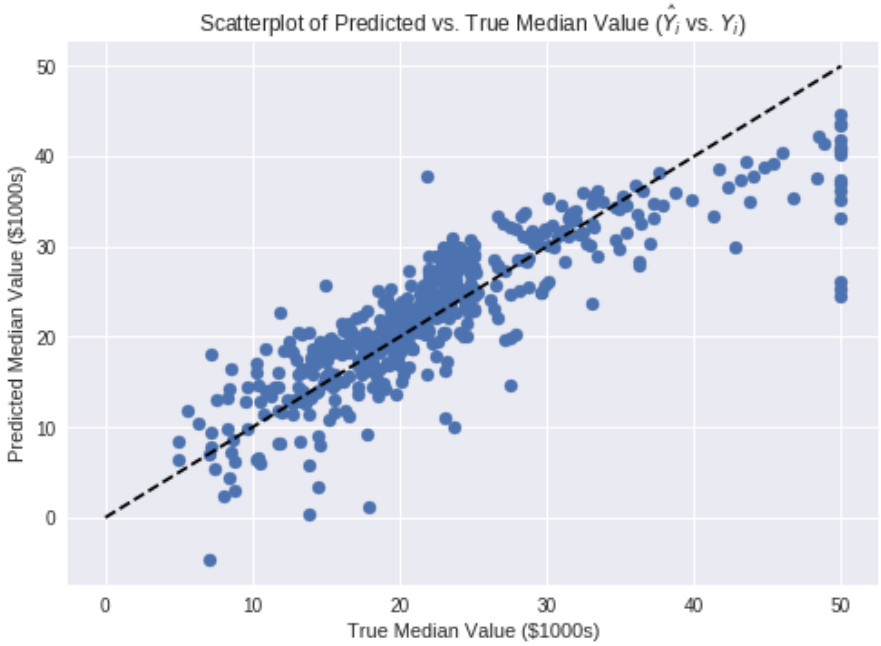
The test RMSEs and training RMSEs of the models from the 10-fold cross validation are displayed below. We observe that the test RMSEs have more variability. For 6 of the iterations, the train RMSE was higher than the test RMSE indicating underfitting of the data. Most of the train RMSEs were around 4.5 but the lowest train RMSE (3.45) had the highest corresponding test RMSE (12.97), indicating that this iteration had overfitting of the data.

```
Test RMSE [ 3.04744921  3.76181913  3.75148053  5.93354231  5.64669077  4.45374875
  3.15392917 12.9759539   5.77319193  3.3106511 ]
Train RMSE [4.83355236 4.78351694 4.81830623 4.55759835 4.61902872 4.72902737
 4.82982623 3.45820725 4.6461054  4.81550033]
```

Plotting fitted values against true values as scatterplots using the best model from k-fold cv:

Figure 2.2 below is a scatterplot of the fitted values against the true values of the median home value using the best model found from k-fold cross validation. The plot looks similar to Figure 2.1 which used the whole dataset to fit the linear regression model.

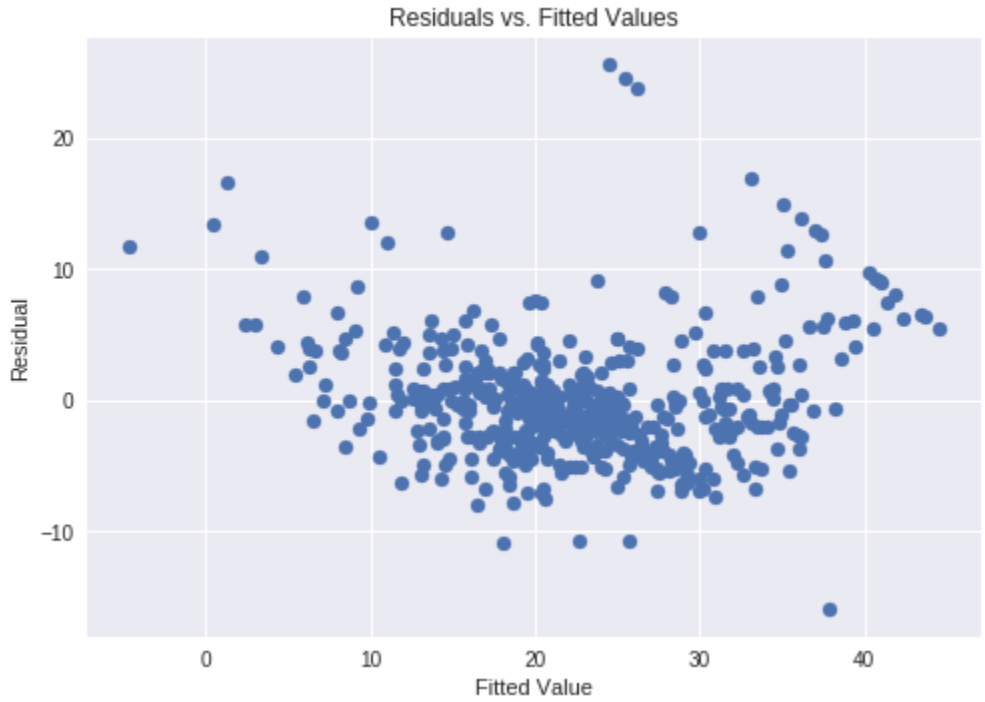
Figure 2.2



Plotting residuals versus fitted values as scatter plots:

We calculated the residuals as `r=true values-fitted values`. A good residuals plot should show even scatter around the 0 line. We observe there may be nonlinearity in the residuals due to the slightly parabolic shape, and that more of the residuals are above the 0 line indicating that the true values tend to be higher than the fitted values.

Figure 2.3



3 Regularization

In this part, we try to control overfitting via regularization of the parameters.

(a) You are asked to try the following regularizations with suitable parameters.

1. Ridge Regularizer: $\min_{\beta} \|Y - X\beta\|^2 + \alpha \|\beta\|_2^2$
2. Lasso Regularizer: $\min_{\beta} \|Y - X\beta\|^2 + \alpha \|\beta\|_1$
3. Elastic Net Regularizer: $\min_{\beta} \|Y - X\beta\|^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2$ (optional)

Optimize over choices of α, λ_1, λ_2 to pick one good model, report the best RMSE obtained via 10-fold cross validation. Compare the values of the estimated coefficients for these regularized good models, with the unregularized best model.

```
using Lasso Regularizer
Optimal Alpha:  12.32846739442066
Minimum Test Rmse:  2.9328401479703987
estimated coefficients:
[-0.          0.02689532 -0.          0.         -0.          0.
  0.         -0.          0.         -0.01055169 -0.          0.00802332
 -0.49851128]
----------------------------------------
using Ridge Regularizer
Optimal Alpha:  0.2477076355991709
Minimum Test Rmse:  3.0455305087426066
estimated coefficients:
[-1.06792851e-01  4.67667289e-02  1.02015739e-02  2.64718615e+00
 -1.53076371e+01  3.82830734e+00 -1.48646580e-03 -1.43934161e+00
  3.00317156e-01 -1.25313023e-02 -9.25546153e-01  9.43969125e-03
 -5.27548902e-01]
----------------------------------------
using Elastic Net Regularizer
Optimal Alpha:  10.47615752789664
Optimal L1 Ratio 0.0
Minimum Test Rmse:  2.798266966318117
estimated coefficients:
[-0.07823032  0.05582952 -0.04440294  0.02606377 -0.00075359  0.16526262
  0.03433185 -0.19961024  0.19091566 -0.01419382 -0.2842187   0.00821149
 -0.63660781]
----------------------------------------
Unregularized Best Model
Minimum Test Rmse: 3.04744921
estimated coefficients:
[-1.05618627e-01  4.91381113e-02  3.21540897e-02  2.51293737e+00
 -1.76280019e+01  3.81765978e+00  1.05792271e-02 -1.43608998e+00
  3.61498721e-01 -1.54712068e-02 -9.12347239e-01  9.85932090e-03
 -5.55092176e-01]
```

A regression model that uses L1 regularization technique is called Lasso Regression and model which uses L2 is called Ridge Regression. Elastic Net produces a regression model that is penalized with both the L1-norm and L2-norm. The consequence of this is to effectively shrink coefficients (like in ridge regression)

and to set some coefficients to zero (as in Lasso). We used L1 ratio which is the Elastic Net mixing parameter, with `0 <= l1_ratio <= 1`. For `l1_ratio = 0` the penalty is an L2 penalty. For `l1_ratio = 1` it is an L1 penalty. For `0 < l1_ratio < 1`, the penalty is a combination of L1 and L2.

To find the optimal alphas, we tested models using alpha between 0.0001 and 100. To find the the optimal L1 and L2 parameters in the Elastic Net model, we tested L1 ratios between 0 and 1. From comparing the regularized models with the best unregularized model via 10-fold cross-validation, we observe that the regularized models provide slightly better performance (slightly lower test RMSEs) but they don't improve the model by much. Comparing the values of the coefficients, the Ridge regularizer gives the closest estimated coefficients to the unregularized model, although in general the coefficients of Ridge are slightly lower in value. The lasso regularizer will shrink the lesser important coefficients to 0, which makes it a good regularizer for feature selection. 9 of the 13 coefficients were shrunk to 0 in the Lasso model; thus, the only important features selected by Lasso are ZN, TAX, B, and LSTAT. The Elastic Net coefficients in general are smaller than the unregularized coefficients, but not as small as the Lasso coefficients.

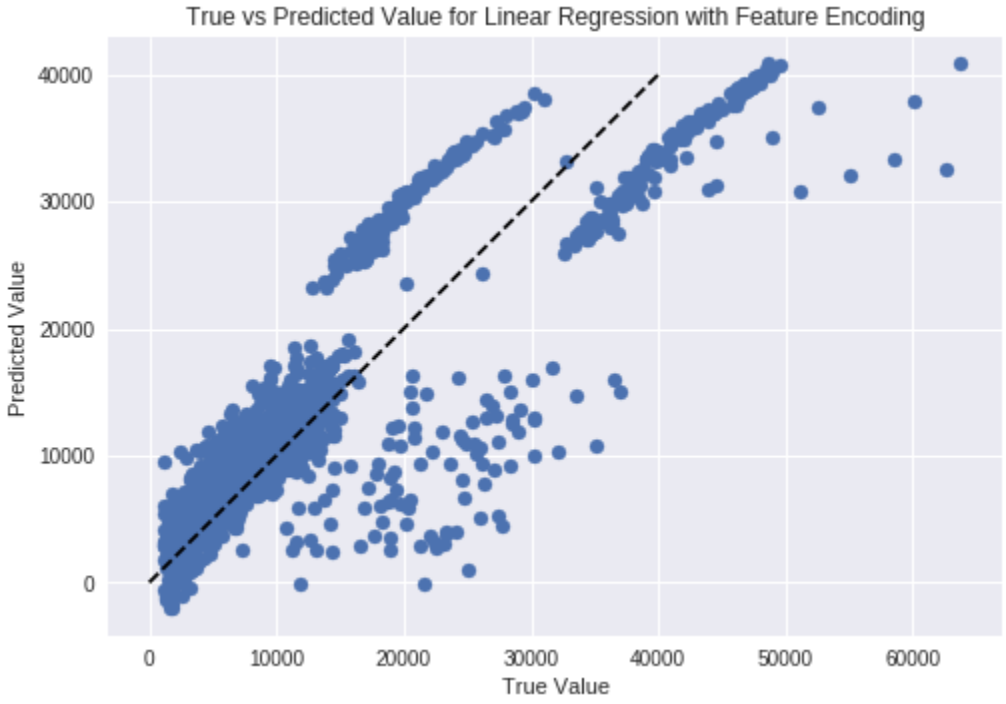
Dataset 3: Car Insurance

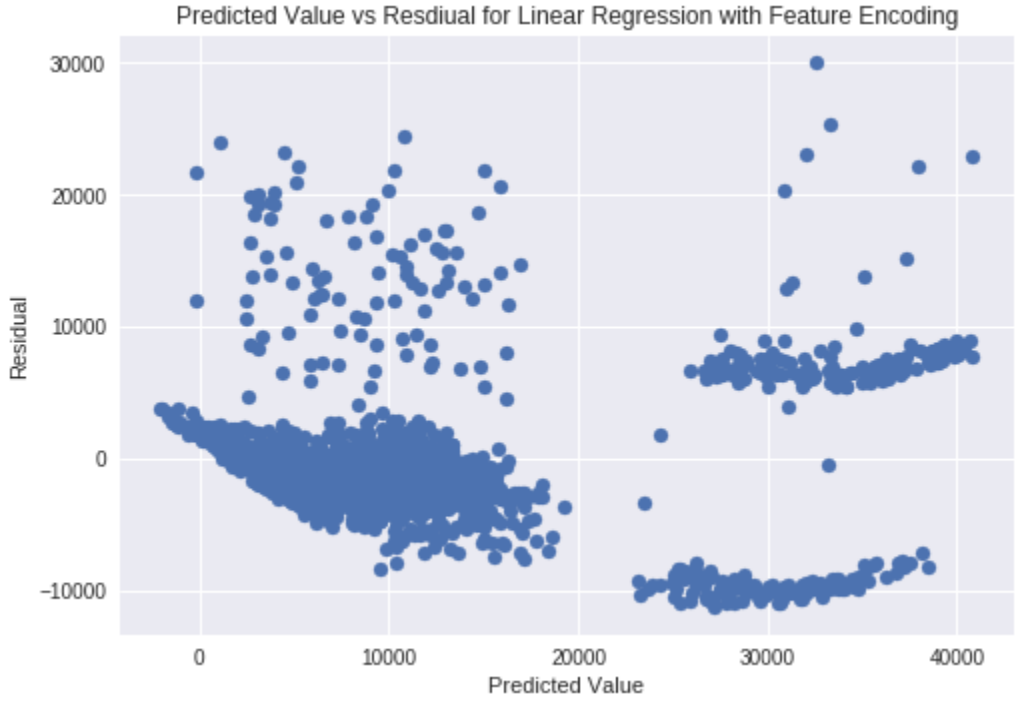
This is one car insurance dataset. We try to train models to predict user's car insurance charges given the 6 features from the user's profile. Each row contains one user's 6 features and car insurance charges. ft1,ft2 and ft3 are numerical features, ft4 and ft5 and ft6 are categorical features.

1. Feature Preprocessing

a. Feature Encoding

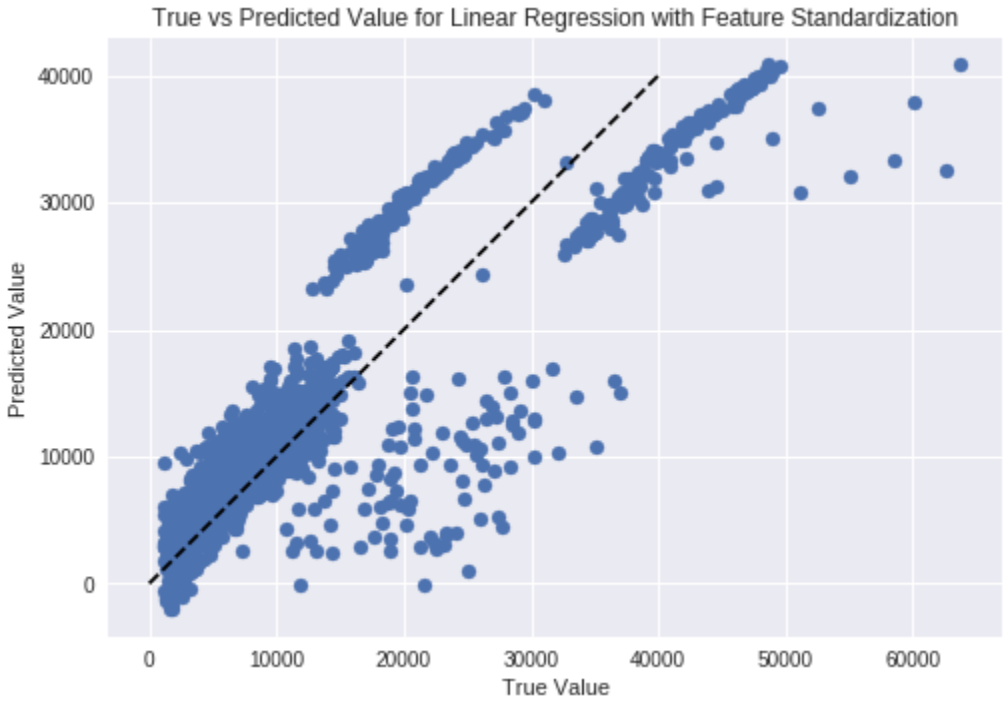
For the one-hot feature encoded vector, we achieved an average training RSME of 6038.42 and an average test RSME of 6080.28 with 10 fold cross validation.

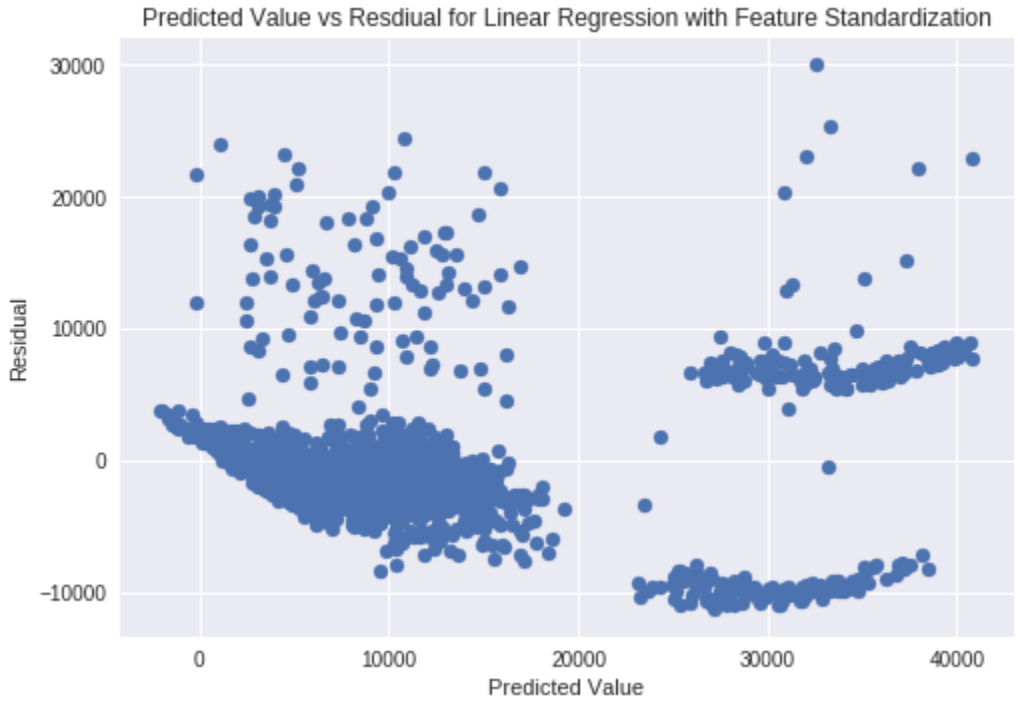




b. Standardization

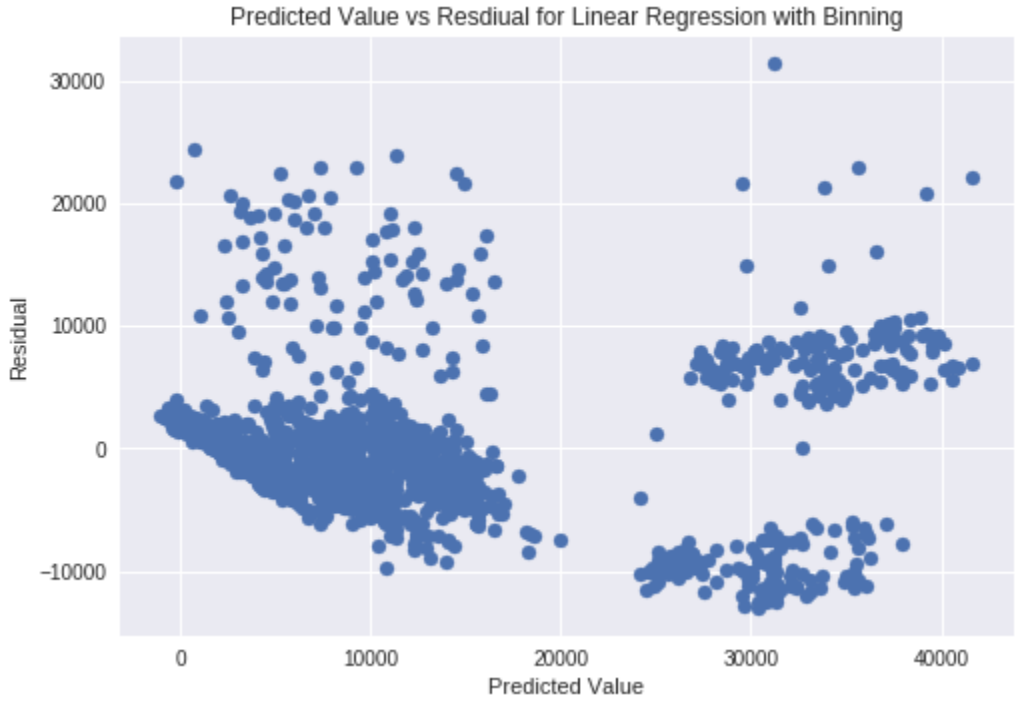
For the standardardized one-hot feature encoded vector, we achieved an average training RSME of 6038.42 and an average test RSME of 6080.28 with 10 fold cross validation. These are the same as for the unstandardized version, which is to be expected, as standard linear regression models are not affected by feature scaling.





c. Binning

For the one-hot feature encoded vector, we achieved an average training RSME of 6197.17 and an average test RSME of 6235.40 with 10 fold cross validation.



2. Correlation exploration:

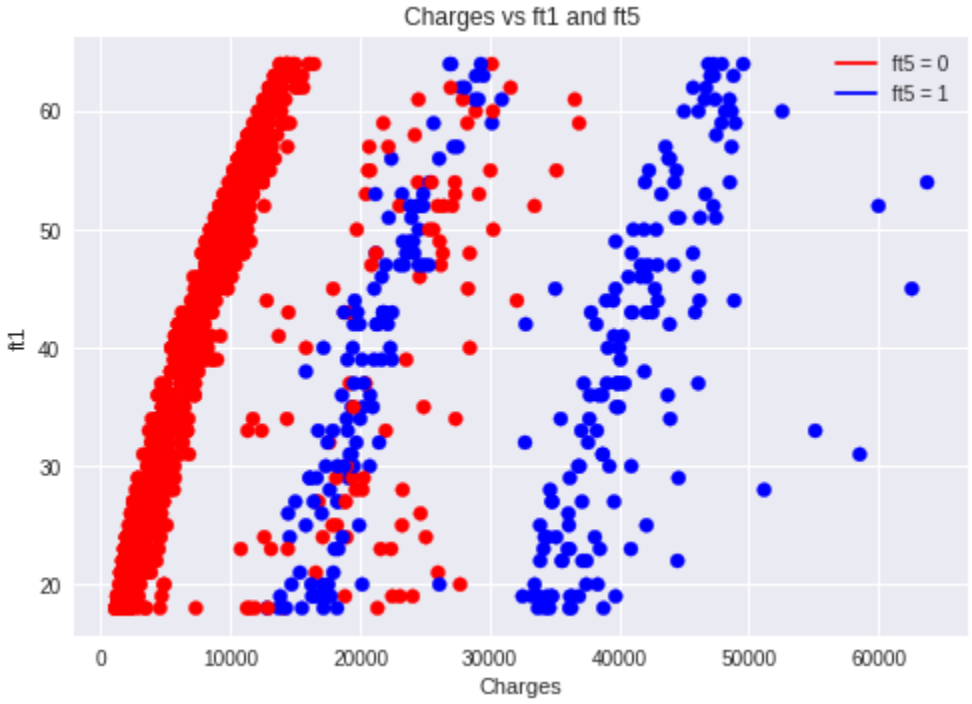
a.

	ft1	ft2	ft3	ft4	ft5	ft6
F score	**131.17**	54.709	6.2060	4.3997	**2177.6**	0.051494
Mutual Info	**1.5038**	0.0741	0.1616	0.1767	**0.3692**	0.0767

For both F score and mutual info score, ft5 and ft1 have the top two scores, suggesting that they are the two most important variables.

b.

c.

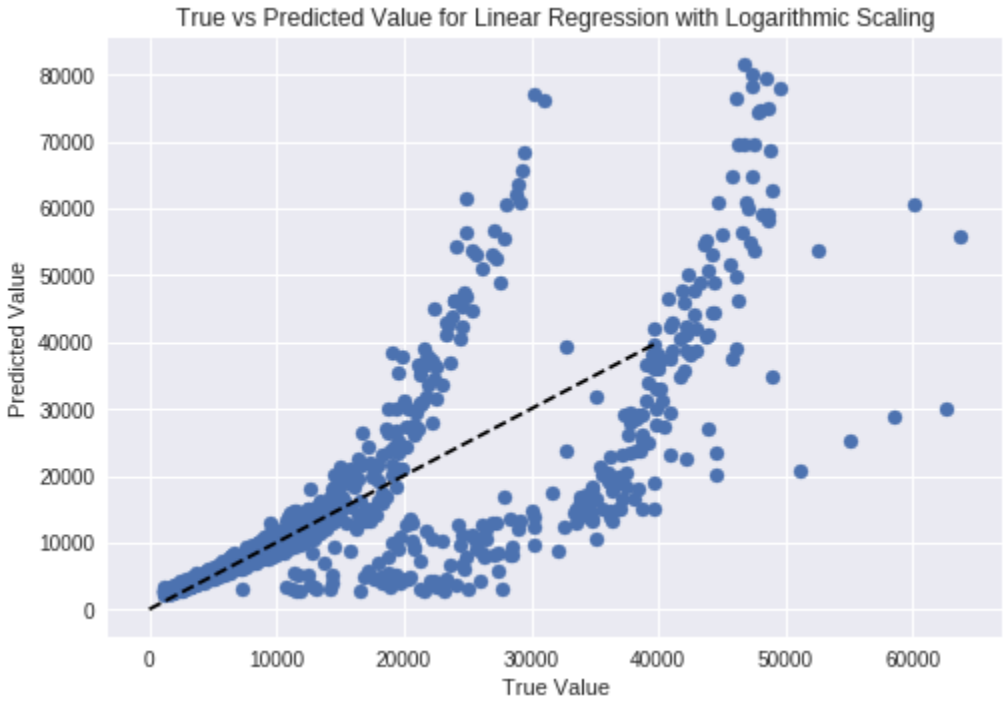


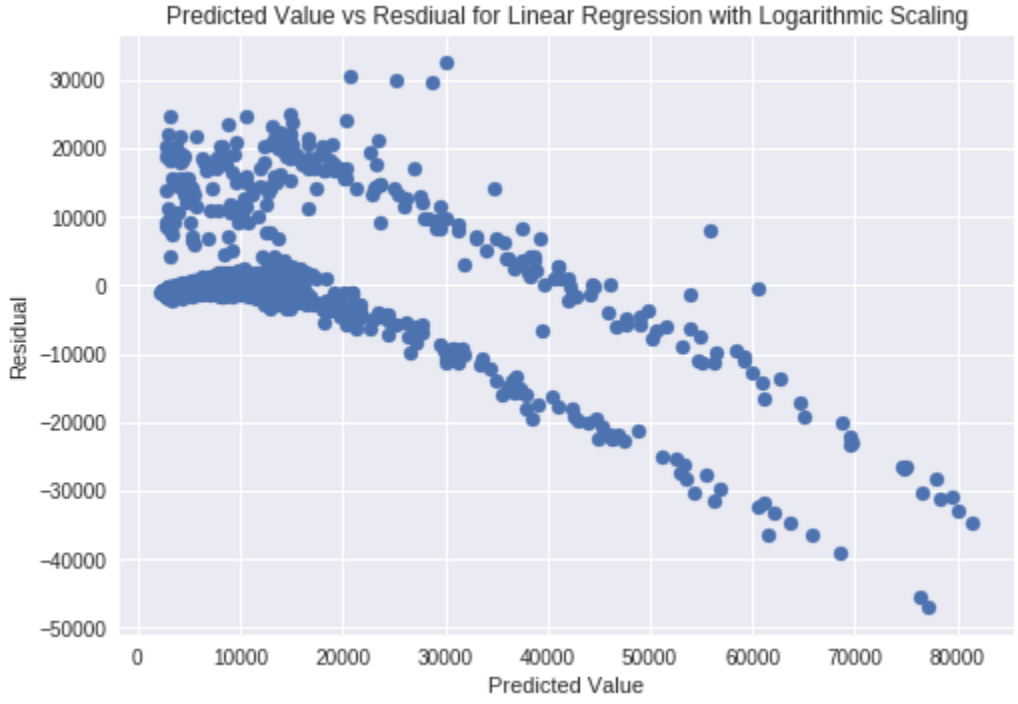
3. Modify the target variable

As we have seen so far, the target variable: charges (y) spans a wide range, so here instead of fitting the original value, we consider fitting log(y). Note here instead of calculating the difference between

predicted value (log(y)predict) and transformed target values (log y), we calculate the difference between exp(log(y)predict) and y to set up a fair comparison.

a. The Feature Encoding method was chosen for feature preprocessing and a linear regression model was trained on the new target. we achieved an average training RSME of 8362.05 and an average test RSME of 8410.46 with 10 fold cross validation, which were greater than the average training and test RSMEs when y was not log transformed. The performance didn't improve after taking log of y.





b.

	ft1	ft2	ft3	ft4	ft5	ft6
F score	**515.98**	23.936	35.705	0.04237	**1062.12**	2.4391
Mutual Info	**1.5029**	0.06776	0.1610	0.1763	**0.3694**	0.07753

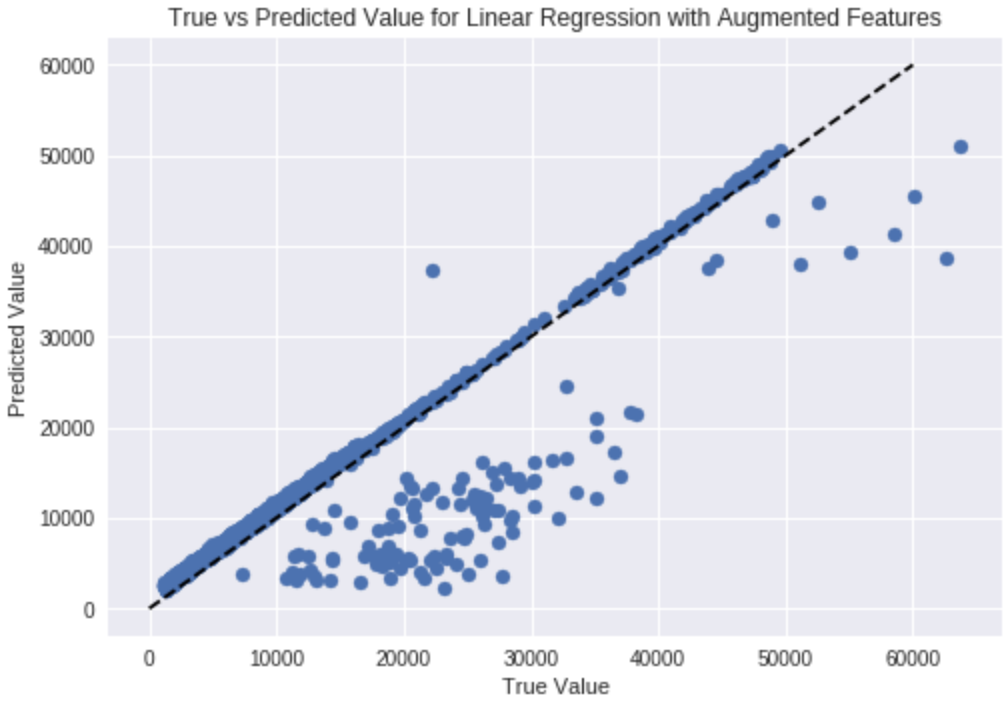
Transforming the charges did not change which features were most important.

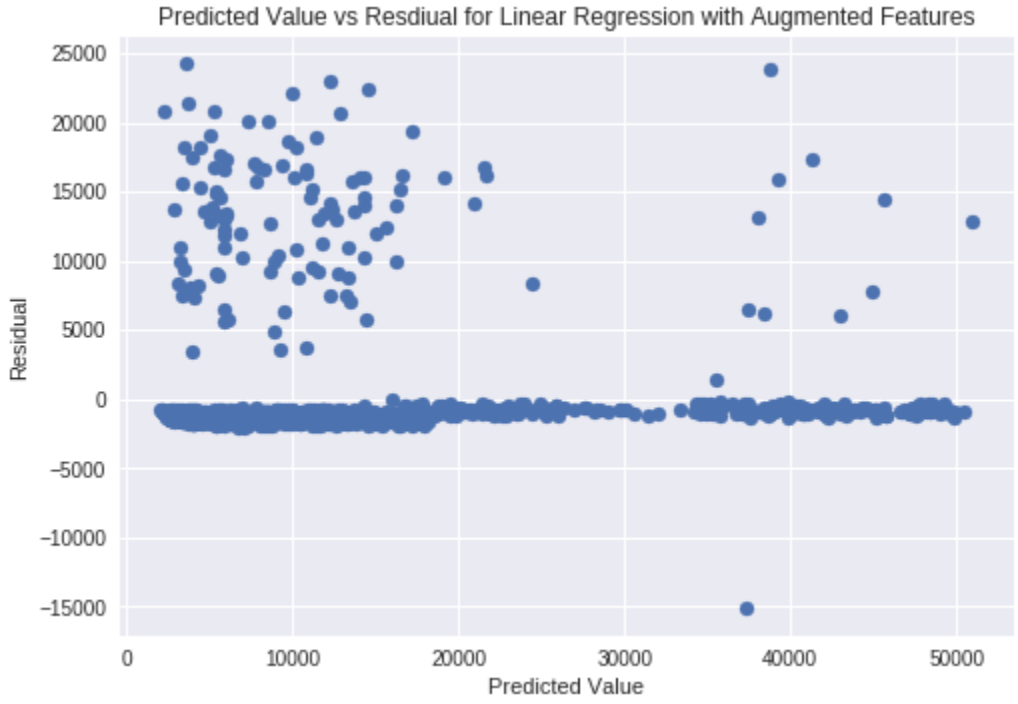
4. Bonus questions:

a. We attempted to improve the results using improved feature encoding. We implemented three types of new features in addition to the previous features: binning, polynomials of one variable, and products of two variables. The choice of variables were based on the correlation experiments from part 2. The new features are as follows:

Feature	Description
ft7	ft2*ft5
ft8	ft2*(1-ft5)
ft9	ft1*ft5
ft10	ft1*(1-ft5)
ft11	$(ft1)^2$
ft12	ft2 < 30
ft13	ft12*ft5

With these features, we achieved an average training RSME of 4360.07 and an average test RSME of 4355.16. From these results and the graph of the model trained on the entire dataset, these features seem to be a definite improvement.





Conclusion

From our experiments, we saw that the different models have different strengths. When the data features are mostly numerical, and the underlying relationship between the features and the desired output is linear, simple Linear Regression models perform well, as they are quick to train and generalize well to new data. Even in the case where the relationship between the data and the desired quantity is not strictly linear, linear models can perform well if augmented features are added, but this requires careful manual inspection of the data, and may not generalize well. For handling categorical data in linear models, one hot encoding performs the best, as scalar encoding depends heavily on the order in which the categories are numbered for a linear classifier.

If data the is not linear, or the data contains a lot categorical features, neural networks, random forests, or KNN are likely better options. These models can learn nonlinear relationships, and can better handle categorical data in both scalar and one hot encoding. However, unlike Linear Regression, neural networks and KNN generally required feature standardization to perform effectively. Additionally, as they are more expressive models, all three methods are more prone to overfitting than linear models. Also, these models may be more computationally intensive to train, for neural networks and random forests, and test, for random forests and KNN.