1. BUILDING A MODEL

1.1 Feature Engineering

1.1.a correlation and Heatmap

	Property1	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Property2
Property1	1.000000000	-0.008643927	0.19010902	0.12603105	0.08013663	-0.27742143	-0.35988428	-0.25651585	-0.38223602	0.5335221	0.2567751	0.70803388	0.5334042	-0.2744400
Feature1	-0.008643927	1.000000000	0.29652329	0.66884714	0.18826618	0.10087854	0.13919637	0.12766922	0.13116816	0.1224307	0.3081890	-0.04242538	0.1223664	0.2808083
Feature2	0.190109018	0.296523288	1.00000000	0.80800999	0.57243386	-0.07474680	0.09647261	0.04164351	0.07379965	0.6402814	0.4068780	0.26511839	0.6405854	0.1322982
Feature3	0.126031046	0.668847142	0.80800999	1.00000000	0.50144507	0.01363211	0.14140601	0.10433704	0.11725678	0.4924659	0.4870985	0.14453752	0.4925624	0.2495971
Feature4	0.080136635	0.188266177	0.57243386	0.50144507	1.00000000	0.03068175	0.08698502	0.08257883	0.07450870	0.3483005	0.1994940	0.10807018	0.3484787	0.1048616
Feature5	-0.277421433	0.100878542	-0.07474680	0.01363211	0.03068175	1.00000000	0.49377759	0.86074653	0.53152339	-0.4597172	-0.7358977	-0.53855695	-0.4595582	0.4348175
Feature6	-0.359884275	0.139196366	0.09647261	0.14140601	0.08698502	0.49377759	1.00000000	0.82474432	0.95130648	-0.6445469	-0.3012249	-0.76828820	-0.6443112	0.4136066
Feature7	-0.256515850	0.127669225	0.04164351	0.10433704	0.08257883	0.86074653	0.82474432	1.00000000	0.83783219	-0.5847592	-0.5917812	-0.63828022	-0.5844831	0.4566563
Feature8	-0.382236018	0.131168158	0.07379965	0.11725678	0.07450870	0.53152339	0.95130648	0.83783219	1.00000000	-0.7000117	-0.3481803	-0.78978323	-0.6997758	0.4560742
Feature9	0.533522118	0.122430691	0.64028138	0.49246594	0.34830045	-0.45971719	-0.64454694	-0.58475916	-0.70001168	1.0000000	0.5629895	0.83120311	0.9999960	-0.2764386
Feature10	0.256775128	0.308188990	0.40687796	0.48709852	0.19949403	-0.73589774	-0.30122489	-0.59178121	-0.34818025	0.5629895	1.0000000	0.46645068	0.5630002	-0.1716531
Feature11	0.708033875	-0.042425378	0.26511839	0.14453752	0.10807018	-0.53855695	-0.76828820	-0.63828022	-0.78978323	0.8312031	0.4664507	1.00000000	0.8312265	-0.3913364
Feature12	0.533404172	0.122366398	0.64058536	0.49256238	0.34847871	-0.45955823	-0.64431116	-0.58448310	-0.69977577	0.9999960	0.5630002	0.83122649	1.0000000	-0.2768262
Property2	-0.274440024	0.280808264	0.13229815	0.24959712	0.10486155	0.43481747	0.41360657	0.45665631	0.45607424	-0.2764386	-0.1716531	-0.39133644	-0.2768262	1.0000000

Table 1.1.a: Correlation Matrix

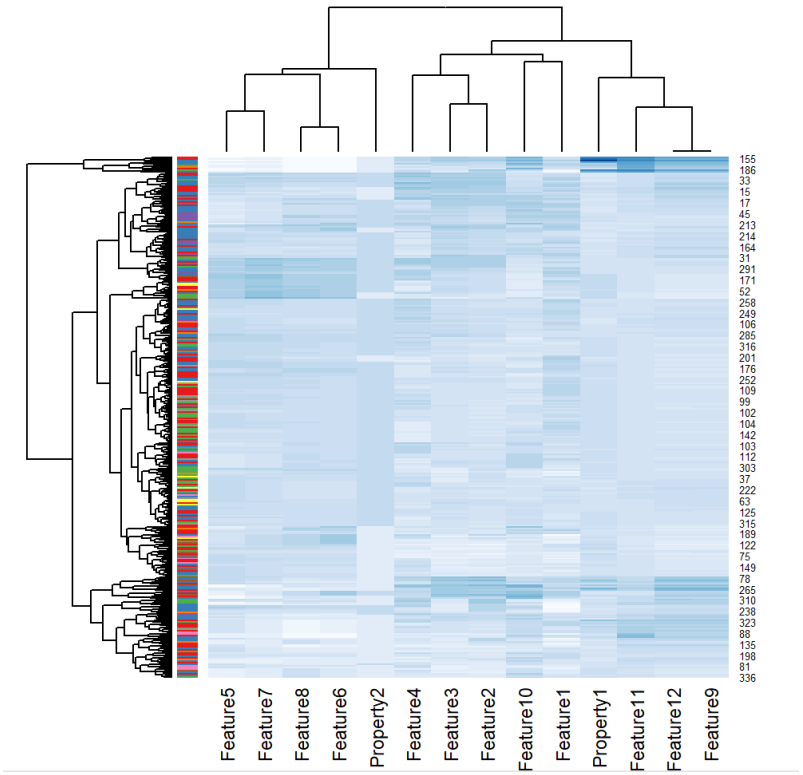

Figure 1.1.a: Heatmap dendrogram

1.1.b VIP

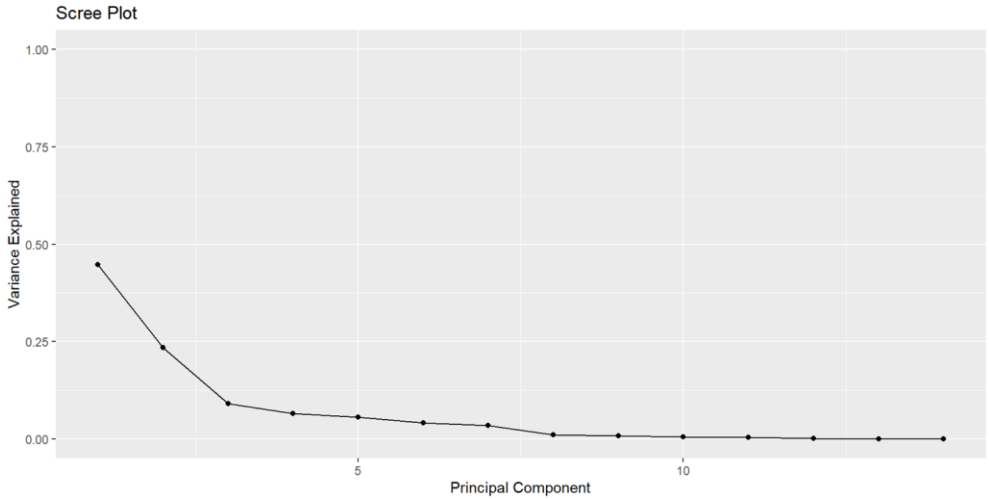

Figure 1.1.b.1: Scree plot demonstrating the data variance

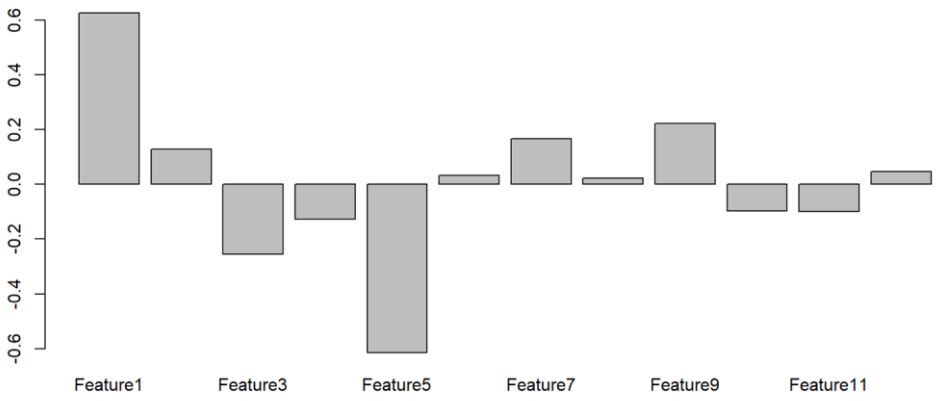

Figure 1.1.b.2: Variable Importance Projection For Property1

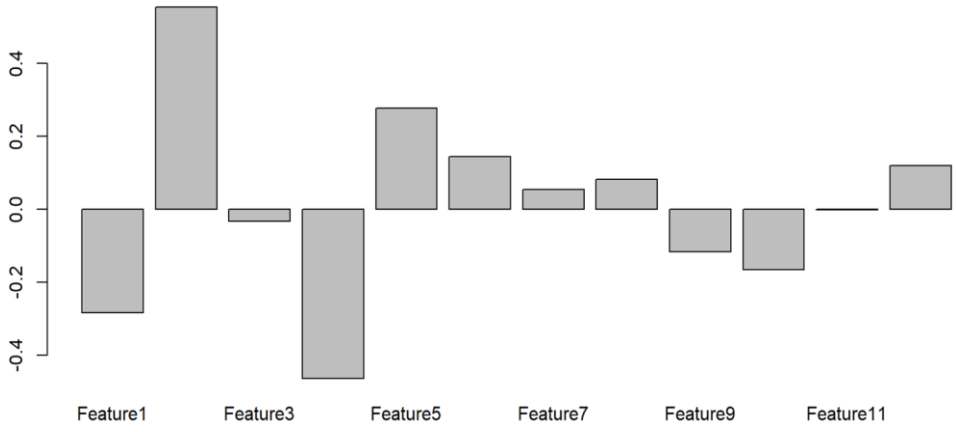

Figure 1.1.b.3: PCA Variable Importance Projection For Property2

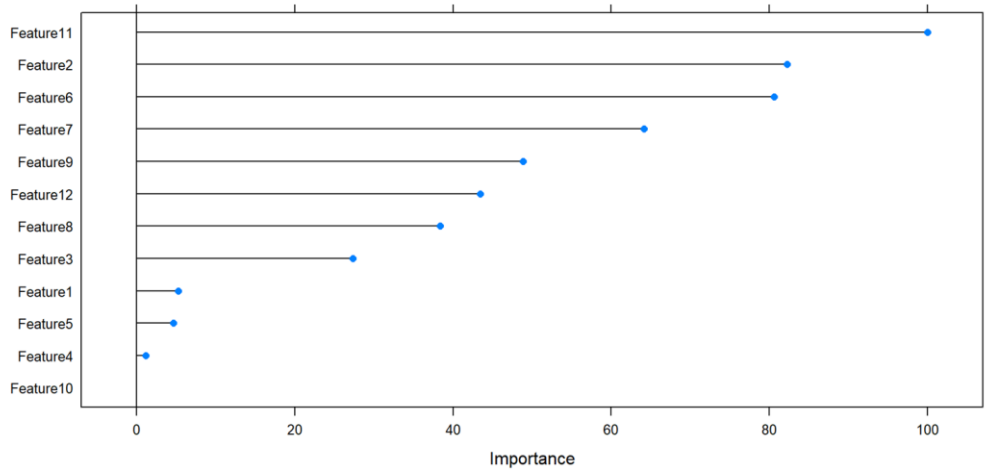

Figure 1.1.b.4: Bagging Variable Importance Projection For Property2

1.1.c Dealing with outliers

The outlier treatment is performed in such a way that the values above 100% quartile range were replaced with the highest quartile value and below the 5% quartile range were replaced with the lowest quartile value.

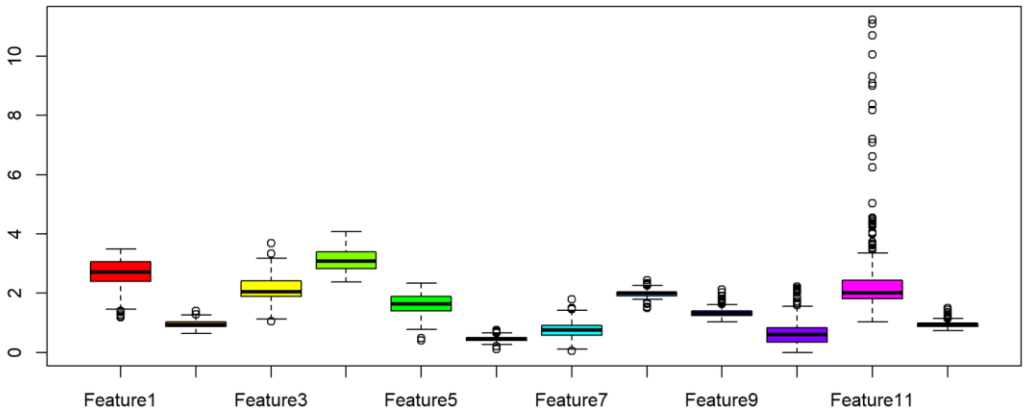

Figure 1.1.c.1: Data Spread before outliers treatment

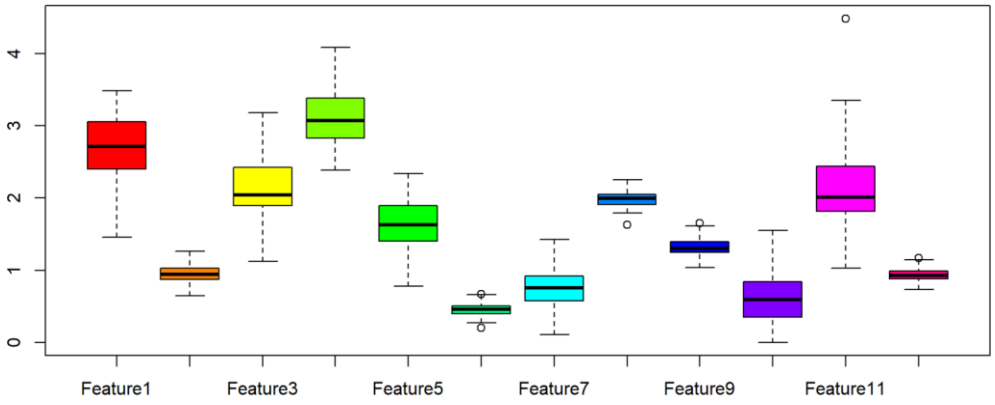

Figure 1.1.c.2: Data Spread after outliers treatment

1.1.d Feature Selection

From comparing the correlation matrix, heatmap dendrogram, PCA VIP and Bagging VIP the following features are selected for building a model:

Property1: Feature5, 8, 9, 10, 11, 12- For the reason that 1, 4, 3, 2 are having least correlation with the response variable; the pairs 5/7, 6/8, 9/12 have the strongest correlation so one among them were selected; as per the VIP 1, 5, 11 are considered important

Property2: Feature1, 5, 8, 9, 11, 12- For the reason that 10, 4, 2 are having least correlation with the response variable; the pairs 5/7, 6/8, 9/12 have the strongest correlation so one among them were selected; as per the VIP 1, 5, 11 are considered important

1.2 Models of the response variable- Property2

1.2.a Support Vector Machine

```
               Accuracy : 0.9893
                 95% CI : (0.969, 0.9978)
    No Information Rate : 0.6357
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9768

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9804
            Specificity : 0.9944
         Pos Pred Value : 0.9901
         Neg Pred Value : 0.9888
             Prevalence : 0.3643
         Detection Rate : 0.3571
   Detection Prevalence : 0.3607
      Balanced Accuracy : 0.9874

       'Positive' Class : 0
```

Figure 1.2.a.1: SVM Model Metrics- Train Data

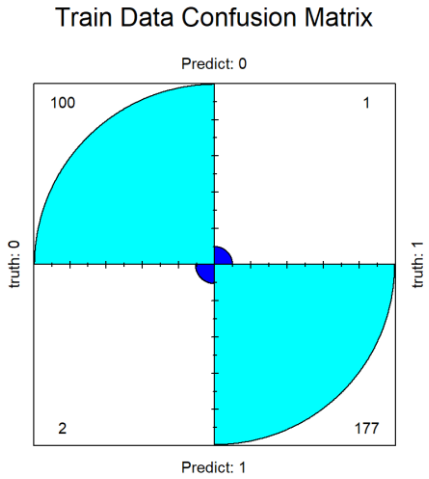

Figure 1.2.a.2: SVM Confusion Matrix- Train Data

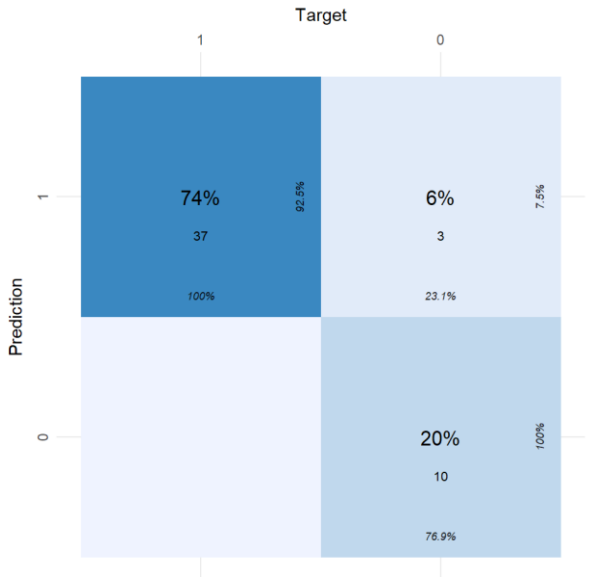

Figure 1.2.a.3: SVM Confusion Matrix- Test Data

1.2.b Bagging

```
Bagged CART

280 samples
 12 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 252, 252, 252, 251, 252, 253, ...
Resampling results:

  Accuracy   Kappa
  0.9325032  0.8540447
```

Figure 1.2.b.1: Model Metrics- Train Data

```
> with(train_data_P2,table(tree.pred_train,Property2))
                Property2
tree.pred_train   0   1
              0 101   0
              1   1 178
> tree.pred_test=predict(train.bagg,test_data_P2)
> with(test_data_P2,table(tree.pred_test,Property2))
               Property2
tree.pred_test  0  1
             0 12  0
             1  1 37
```

Figure 1.2.b.1: Bagging Truth Table

1.2.c Random Forest with Boosting

```
Resampling results across tuning parameters:

  interaction.depth  n.trees  Accuracy   Kappa
  1                   50      0.9355546  0.8565982
  1                  100      0.9321064  0.8508066
  1                  150      0.9388706  0.8663372
  2                   50      0.9391261  0.8652686
  2                  100      0.9424421  0.8742697
  2                  150      0.9460135  0.8815546
  3                   50      0.9497172  0.8886491
  3                  100      0.9497172  0.8889950
  3                  150      0.9425652  0.8743586

Tuning parameter 'shrinkage' was held constant at a value of 0.1

Tuning parameter 'n.minobsinnode' was held constant at a value of 10
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were n.trees = 50,
 interaction.depth = 3, shrinkage = 0.1 and n.minobsinnode = 10.
```

Figure 1.2.c.1: Random Forest Model Metric

```
> with(train_data_P2,table(tree.pred_train,Property2))
              Property2
tree.pred_train   0   1
              0 100   3
              1   2 175
> tree.pred_test=predict(train.gbm,test_data_P2)
> with(test_data_P2,table(tree.pred_test,Property2))
              Property2
tree.pred_test  0  1
             0 11  0
             1  2 37
```

Figure 1.2.c.2: Random Forest Truth Table

1.2.d Logistic Regression

```
Call:  glm(formula = train_data_P2$Property2 ~ ., family = binomial,
    data = train_data_P2)

Coefficients:
(Intercept)     Feature1     Feature2     Feature3     Feature4
 -2.706e+01   -5.694e-02    2.363e+01    1.333e+00   -4.947e-01
   Feature5     Feature6     Feature7     Feature8     Feature9
  6.220e+00   -4.947e+01   -7.858e+00    1.816e+01    1.317e+03
  Feature10    Feature11    Feature12
  2.837e-01   -1.150e+01   -1.850e+03

Degrees of Freedom: 279 Total (i.e. Null);  267 Residual
Null Deviance:      367.3
Residual Deviance: 192  AIC: 218
```

Figure 1.2.d.1: Logistic Regression Model Parameters

```
                truth
glm.pred_train   0   1
             0  75   5
             1  27 173
```

	truth	
glm.pred_test	0	1
0	8	0
1	5	37

Figure 1.2.d.2: Logistic Regression Truth Table

1.2.e Classification Models Comparison- Property2

By comparing the model metrics to predict 'Property2' the SVM and decision tree with boosting technique (with 98% and 93% accuracy respectively) were observed to be more robust and a great sensitivity and specificity compared to others. The logistic regression is not much precise with a poor specificity. Also noticed that the SVM and boosting models prediction behaviour are not varying with random test and train data. The tables and figures of 1.2.a,1.2.b,1.2.c and 1.2.d are a sure demonstrations of the inferences. Hence we would be utilizing these two models to predict the values of unknowns and whose results are mentioned under section 2.2.

1.3 Models of the response variable- Property1

1.3.a Support Vector Regression

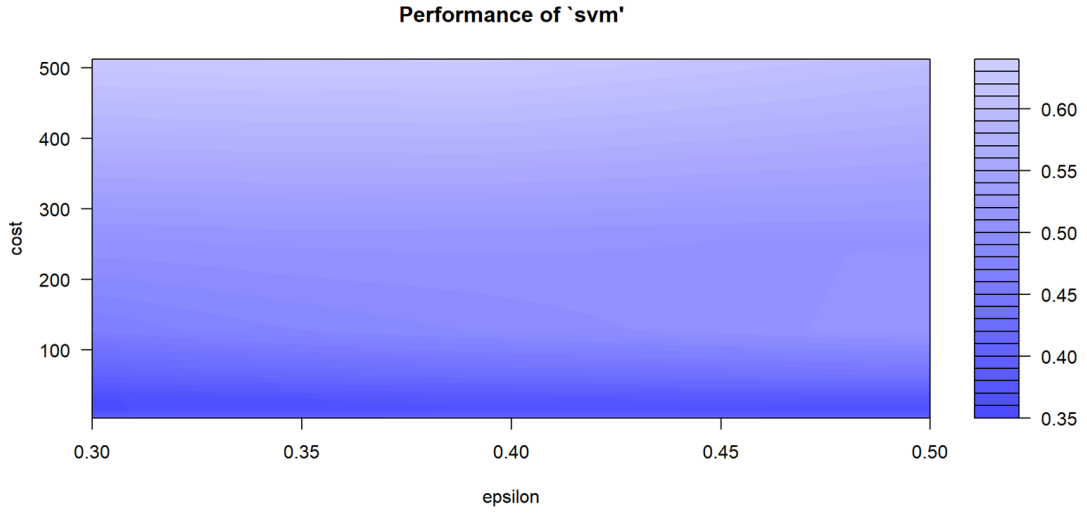

Figure1.3.a.1: SVM Parameter Tuning

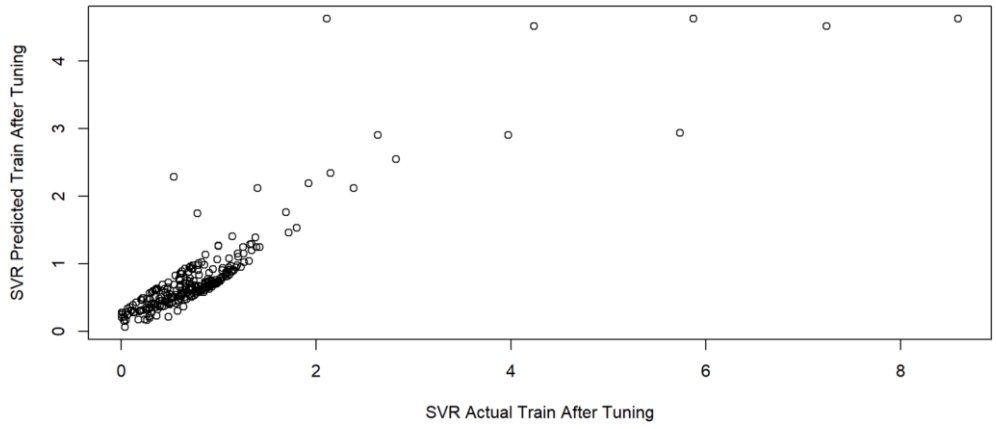

Figure 1.3.a.2: SVR Metric Plot- Train Data

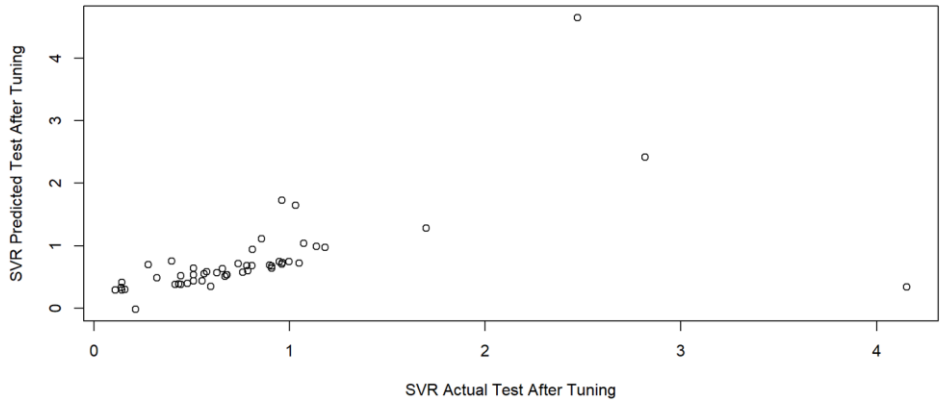

Figure 1.3.a.3: SVR Metric Plot- Test Data

1.3.b Lasso Regression

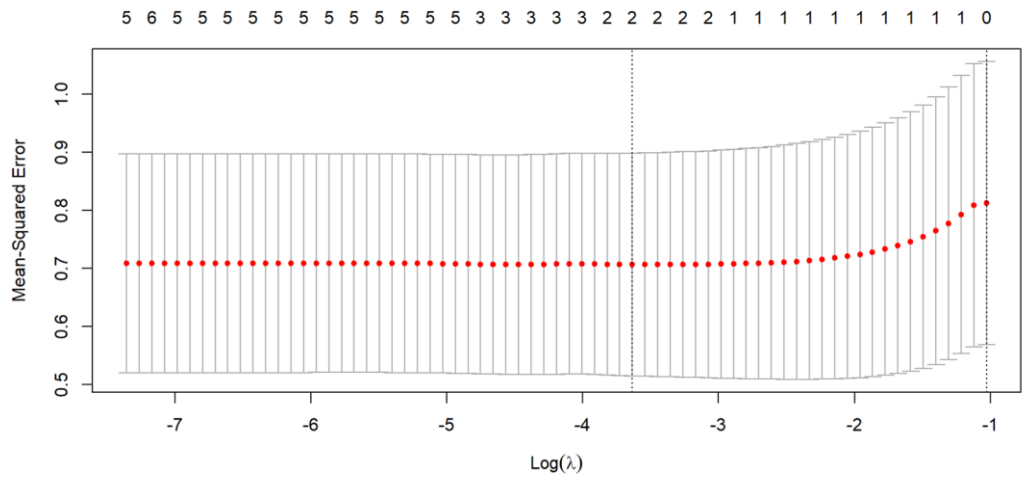

Figure 1.3.b.1: Finding optimal lambda value

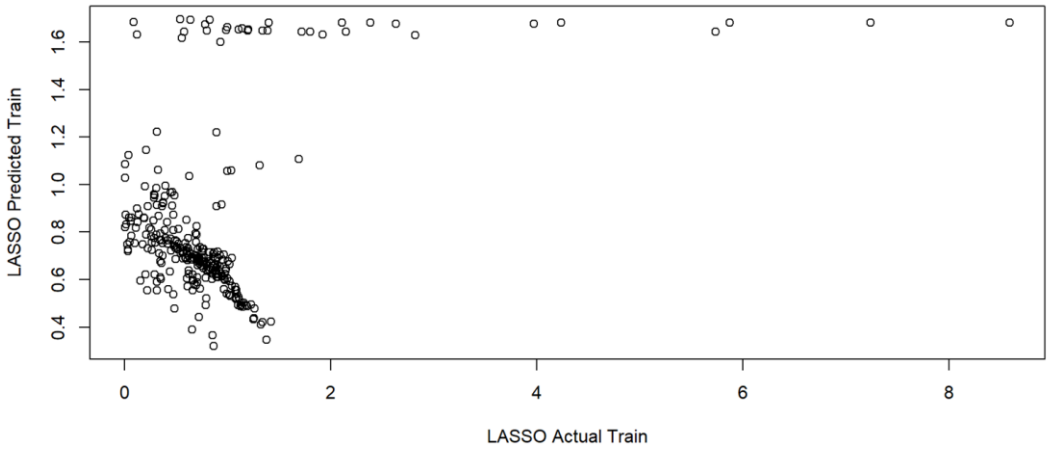

Figure 1.3.b.2: Lasso Metric Plot- Train Data

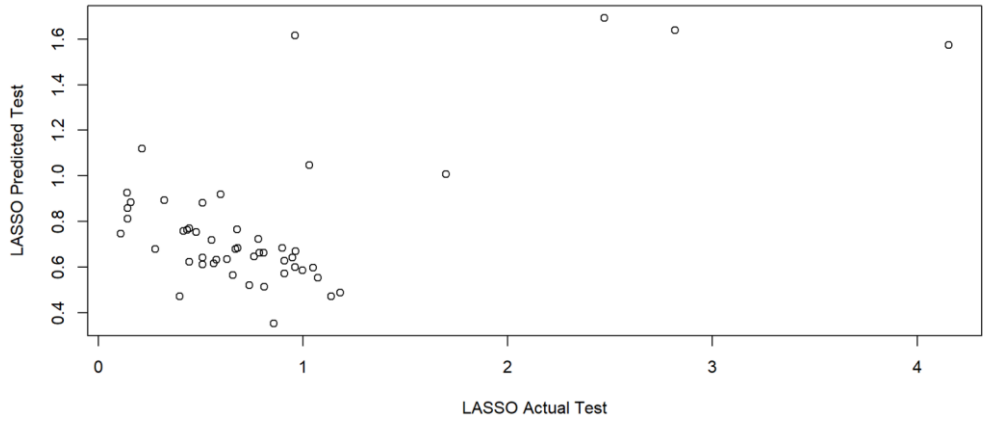

Figure 1.3.b.3: Lasso Metric Plot- Test Data

1.3.c Ridge Regression

Optimal lambda value= 0.01258925

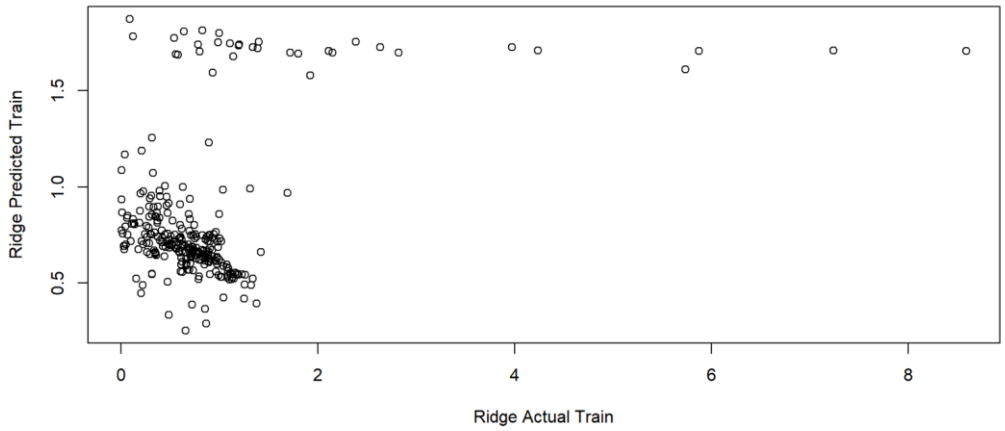

Figure 1.3.c.1: Ridge Metric Plot- Train Data

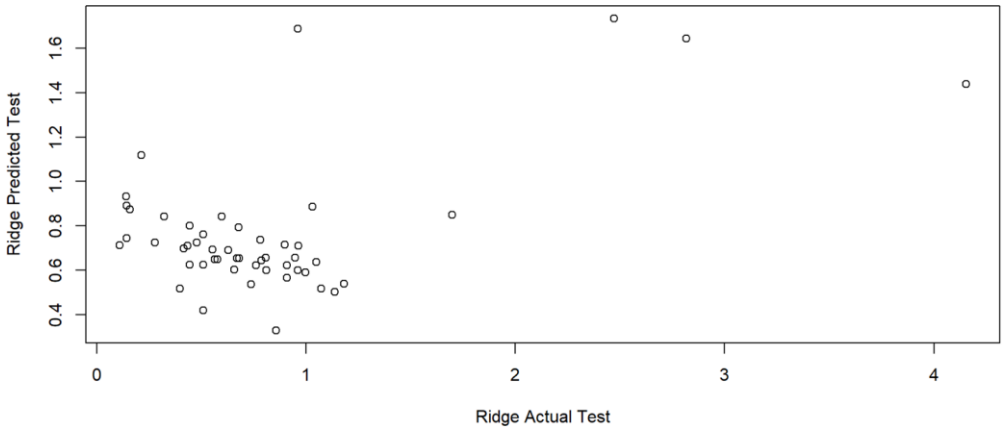

Figure 1.3.c.2: Ridge Metric Plot- Test Data

1.3.d Gaussian Process Regression

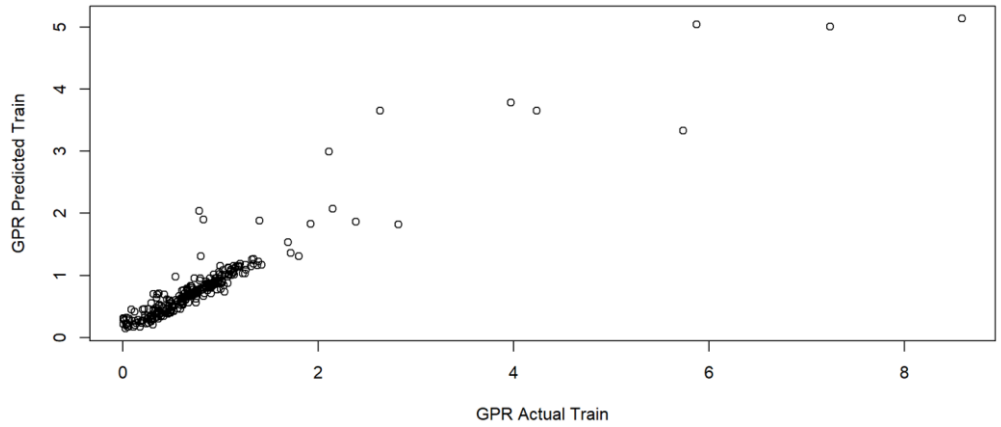

Figure 1.3.d.1: GPR Metric Plot- Train Data

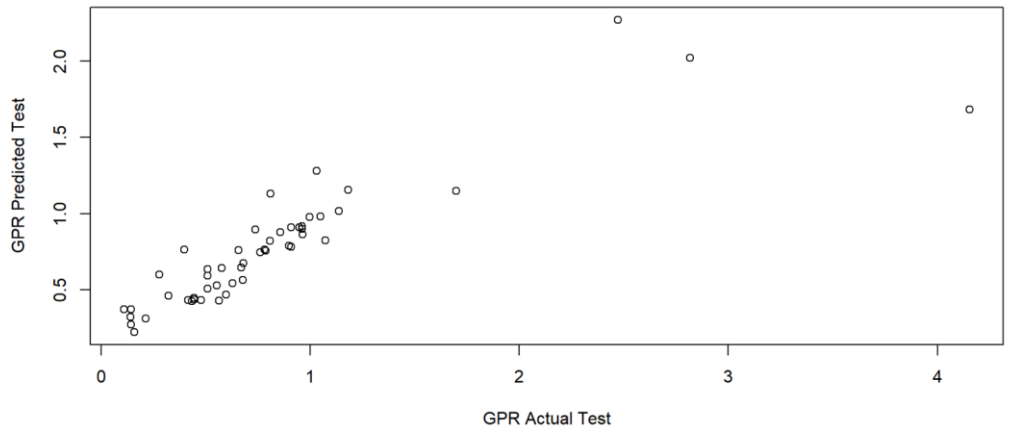

Figure 1.3.d.2: GPR Metric Plot- Test Data

1.3.e Regression Models Comparison- Property1

		SVR	Lasso	Ridge	GPR
TRAIN	RMSE	0.4369	0.8254	0.8202	0.3495
	R-Square	0.7639	0.5103	0.5103	0.8489
TEST	RMSE	0.6636	0.5756	0.5873	0.3981
	R-Square	0.7886	0.5342	0.5342	0.6685

Figure 1.3.e: Accuracy Metric Chart

By comparing the model metrics to predict 'Property1' the SVR and GPR were observed to be more robust compared to others. The train accuracy is better for GPR while the test accuracy is good for SVR. For the SVR the tuning parameters(epsilon and cost) played a very good role to optimise the precision levels. For the successful implementation the concept of grid search is utilised. On the other-hand the tuning parameter lambda for ridge and lasso regression models did not play a vital role in improving the model fit and there-by the prediction. Thus, the SVR and GPR models were utilised to predict the values of unknowns and whose results are mentioned under section 2.1. Also noticed that the model prediction behaviour is not varying much with random test and train data. The tables and figures of 1.3.a,1.3.b,1.3.c,1.3.d and 1.3.e are a sure demonstrations of the inferences.

2. PREDICTING UNKNOWNS

2.1 Property 1 Prediction

	SVR-Predicted Property1
1	0.6969896
2	0.4783642
3	0.2415802
4	1.2464437
5	1.0287165
6	0.3629134
7	0.2317533
8	0.3039831
9	0.5267086
10	1.1850627

Figure 2.1.a: Prediction through SVR Model

	GPR-Predicted Property1
1	0.6364151
2	0.5886749
3	0.3877091
4	0.8247048
5	1.3789815
6	0.4571541
7	0.3257724
8	0.4261612
9	0.5101782
10	0.9691557

Figure 2.1.b: Prediction through GPR Model

2.2 Property 2 Prediction

	SVM-Predicted Property2
1	No
2	No
3	Yes
4	No
5	No
6	Yes
7	Yes
8	Yes
9	No
10	No

Figure 2.2.a: Prediction through SVM Model

	Boosting-Predicted Property2
1	Yes
2	No
3	Yes
4	No
5	No
6	Yes
7	Yes
8	Yes
9	No
10	No

Figure 2.2.b: Prediction through Boosting Model

3. CONCLUSION

The characteristic is to identify the large value of 'Property1' with "yes" as Property2 element. From the original dataset the large value of Property1 typically falls in the range between 2 and 8.5. After combining the results of the Figure 2.1.a, 2.1.b, 2.2.a, 2.2.b, for the "yes" value of Property2 the values of Property1 is within the range from 0 to 0.5. The justification is made by considering the most precise models as per the estimated tuning parameters for regression models and cost factor, epsilon and entropy for the respective classification models. Also, the original dataset has the large Property1 values with Property2 as "no". Therefore it can be assumed that the developed models could not be wrong, rather those are not able to identify the characteristic of the design perspective.