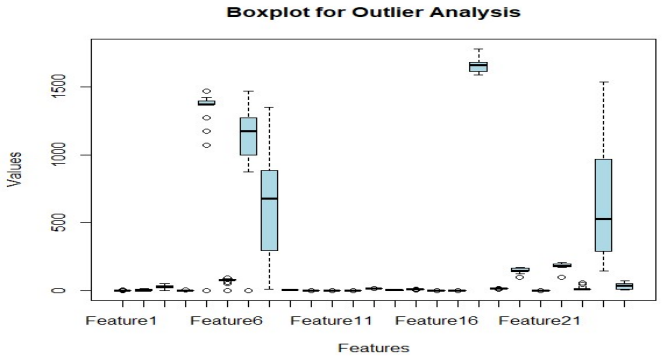
Problem 01

Step 01: Missing data rows were separated from the raw data.

EDA (Exploratory Data Analysis) – Steps 02 to 05 are common to all the regression models used.

Step 02: Scaling of the data: Due to the significant variation in the magnitude of the data for each feature value, I performed scaling. This would also remove bias among the features from models.

Step 03: Outlier Analysis: Plotted the box plot for finding outliers. Based on the criteria of 3 IQR, outliers were removed from the data. This is done to enhance the robustness of the model and take away the disparity due to extreme values.

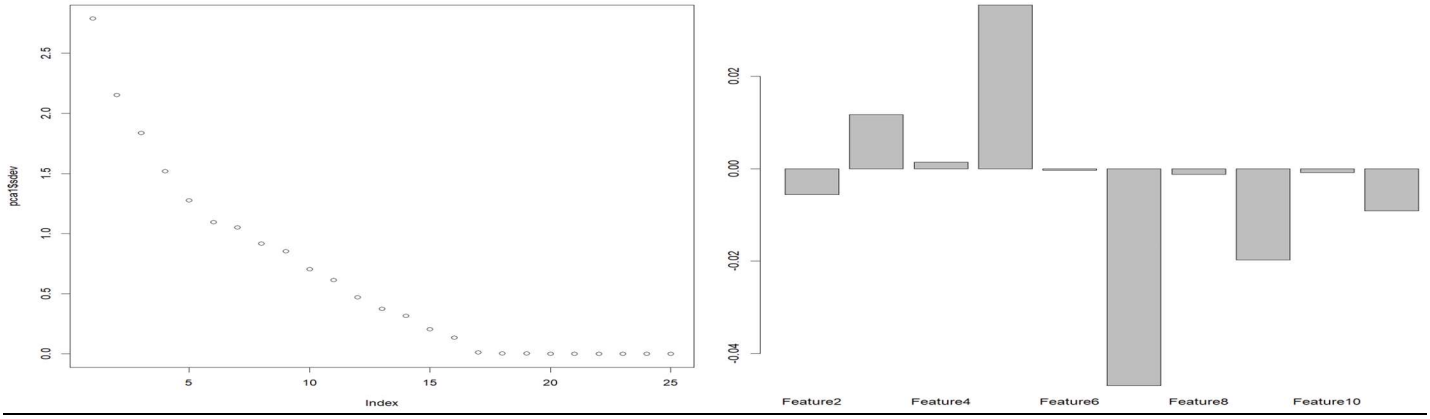


	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Feature13	Feature14	Feature15	Fea
Feature1	1.000000e+00	-0.182131264	-0.212292907	7.146154e-02	0.032227811	0.108607727	-0.10770155	0.021851969	-0.160800619	-0.03332247537	-0.02663870	0.009655895	-1.029062e-01	-0.1126399995	0.100776313	C
Feature2	-1.821313e-01	1.000000000	-0.180955706	-2.546552e-01	0.177911501	-0.016895661	0.32706529	-0.025435305	-0.094212372	-0.08086095535	-0.05471872	-0.002401595	-2.759399e-01	-0.0143964920	0.051480344	-C
Feature3	-2.122929e-01	-0.180955706	1.000000000	-4.126801e-02	0.061659077	0.237564867	-0.19899107	-0.390490307	-0.091890696	0.19629194651	0.16094820	0.203812824	-8.354574e-03	-0.2413568833	-0.080049283	C
Feature4	7.146154e-02	-0.254855211	-0.041268005	1.000000e+00	0.196084432	-0.337422055	-0.14979394	0.158223270	0.361415662	0.71193905396	0.70582850	0.668918977	1.133798e-05	0.2049269993	-0.664512903	C
Feature5	3.222781e-02	0.177911501	0.061659077	1.960844e-01	1.000000000	-0.158520199	-0.11201257	-0.276159384	-0.218263377	0.18356302623	0.27278179	0.293192240	-7.215003e-01	-0.2316710975	-0.006637036	-C
Feature6	1.086077e-01	-0.016895661	0.237564867	-3.374221e-01	-0.158520199	1.000000000	0.04536418	-0.132530839	-0.028051079	-0.14710360318	-0.15460711	-0.173626885	9.732998e-02	-0.0484443246	0.137644709	-C
Feature7	-1.077016e-01	0.327065288	-0.198991069	-1.497939e-01	-0.112012567	0.045364183	1.00000000	-0.079211834	-0.200980180	-0.09022953591	-0.08642774	-0.019536334	-2.345937e-01	-0.2005576401	0.143543592	C
Feature8	2.185197e-02	-0.025435305	-0.390490307	1.582233e-01	-0.276159384	-0.132530839	-0.07921183	1.000000000	0.280191662	0.16796658804	0.14442421	0.105695334	3.049117e-01	0.2634455012	-0.281960022	C
Feature9	-1.608006e-01	-0.094212372	-0.091890696	3.614157e-01	-0.218263377	-0.028051079	-0.20098018	0.280191662	1.000000000	0.43459473471	0.44979904	0.174246343	6.559108e-01	0.9484465964	-0.760980278	-C
Feature10	-3.322475e-02	-0.080860954	0.196291947	7.193905e-01	0.183563026	-0.147103602	-0.09029536	0.167965580	0.434994735	1.00000000000	0.99184726	0.956696842	-1.168980e-01	0.2404358855	-0.894662957	C
Feature11	-2.663070e-02	-0.054716719	0.160948195	7.058285e-01	0.272781791	-0.154607106	-0.08642774	0.144424205	0.449799044	0.99184725893	1.00000000	0.950009143	-1.723424e-01	0.2694663452	-0.894512983	C
Feature12	9.655895e-03	-0.002401595	0.203812824	6.689190e-01	0.293192240	-0.173626885	-0.01953633	0.105695334	0.174246343	0.95669684820	0.95000914	1.000000000	-3.591684e-01	-0.0056222968	-0.759738955	C
Feature13	-1.029062e-01	-0.275939935	-0.008354574	1.133798e-05	-0.721500285	0.097329979	-0.23459372	0.304911747	0.655910835	-0.11688979573	-0.17234237	-0.359168412	1.000000e+00	0.6475483094	-0.225858341	C
Feature14	-1.126400e-01	-0.014364920	-0.241356883	2.049270e-01	-0.231671097	-0.048443246	-0.20055764	0.263445012	0.948465964	0.24043585645	0.26946345	-0.005622968	6.475484e-01	1.0000000000	-0.609852332	-C
Feature15	1.007635e-01	0.051480344	-0.080049283	-6.645129e-01	-0.006637036	0.137644709	0.14354359	-0.281960022	-0.760980278	-0.89466295666	-0.89451298	-0.759738955	-2.258583e-01	-0.6098523332	1.000000000	-C
Feature16	9.408189e-03	-0.173010314	0.370015967	7.308647e-01	-0.353967409	-0.007749884	0.07770610	0.147747815	-0.332791976	0.75473974743	0.74847465	0.349691956	3.917541e-02	-0.5135210914	-0.075466918	1

Step 04: Pair-wise Correlation: I can narrow down the features based on the direct or inverse correlation of one feature with the other, taking 80% as the threshold correlation value. Feature 10 is positively correlated with Features 11 & 12 and is negatively correlated with Features 20, 21 & 22. Feature 17 is positively correlated with Feature 23 and Feature 24 is positively correlated to Feature 25. Hence, I dropped Feature 11, 12, 20, 21, 22, 23, 25.

Step 05: Principal Component Analysis: Using PCA reduces the dimensionality of the dataset by condensing features into fewer principal components.

PCA Elbow Plot: We can observe that approximately 6 principal components give us 90% variance.



PCA VIP PLOT: ***this plot helped me to identify the features with higher importance for the analysis. Based on the height of the bar, Feature 2, 3, 5,7,9 & 11 are the important features.***

Step 06: I split the training to testing data in the ratio of 80:20 for understanding accuracy of the training and testing data.
Step 07: Respective model based on the property type was built based on the type and parameters required.
Step 08: Calculated RMSE of training and testing data and R-squared value.

Property 1: Model accuracy

Model	Train RMSE	Test RMSE	R-squared

Multiple Linear Regression	137.1978	131.3393	0.883698
Ridge Regression	206.3866	143.824	0.736818
Random Forest	149.2162	104.681	0.86243

Random forest has the lowest Testing RMSE value, indicating least error on unseen dataset as compared to the other models. It also has relatively high R-squared value explaining a major portion of variance in the data.

Property 2: Model accuracy

Model	Train RMSE	Test RMSE	R-squared
Lasso regression	14.17616	13.57231	0.4997178
Gaussian Process regression	12.16228	13.31031	0.6317625
Random forest	9.905548	11.48654	0.7557386

Random forest has the lowest Testing RMSE value proving best predictive accuracy on unseen data.
It also has the highest R-squared value, explaining greater portions of variance in the target variable.

Property 3: Model evaluation metrics

	Accuracy		Sensitivity		Specificity	
Model	Training	Test	Training	Test	Training	Test
Decision tree	48.44%	87.50%	50%	100%	47.62%	81.82%
Logistic Regression	68.75%	50%	74.07%	42.86%	64.86%	55.56%
Random forest	93.75%	56.25%	96.77%	50%	90.91%	62.50%

Based on the metrics a good balance between test accuracy, sensitivity and specificity while not overfitting to the training data will be the best model. Logistic Regression is the best fit as it doesn't exhibit as much overfitting.

<u>Reason for choosing techniques for property 1:</u>
1. MLR is a good baseline model for regression tasks. It presumes if there is linear relationship between descriptors and the target variables.
2. Ridge regression penalizes the large coefficients thereby preventing overfitting or multicollinearity.
3. Random Forest is robust to outliers and can handle complex relationships.

Overall, the three models incorporate different advantages based on the different techniques and hence cover different issues with the data to fetch the best results from the data. All three methods differ in their interpretability and computational complexity.
Thus, I believe that the selection is comprehensive, and I am confident that these techniques are sufficient.

<u>Reason for choosing techniques for property 2:</u>
1. Lasso regression L1 regularization, which can shrink coefficients to zero, effectively performing selection of the feature.
2. GPR can capture complex relationships. It is good with the measurement of uncertainty.
3. Random forest has ability to handle non-linear relationships, robust against outliers.

Overall, the three techniques encompass a diverse set of modeling techniques. This range of accuracy gives me confidence in my predictive modelling, as it helps me to assess both linear and non-linear relationships assumptions.

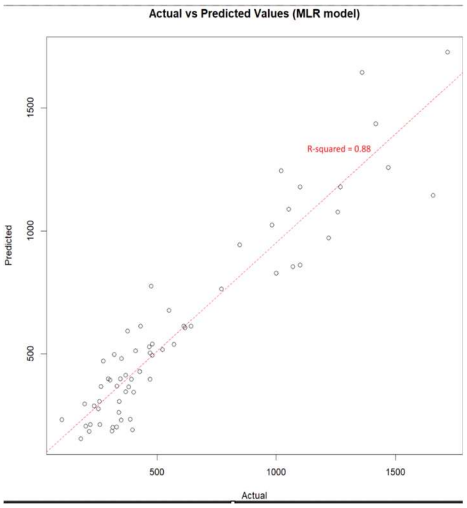
<u>Reason for choosing techniques for property 3:</u>
1. A Decision tree is easier to interpret and can handle non-linear relationship. I chose it for interpretability.
2. Logistic regression is a parametric approach that assumes linear relationship. It's appropriate for linear boundary.
3. Random forest has ability to handle non-linear relationships, robust against outliers and controls overfitting.

In conclusion, I am confident with the selection of the models I chose as these techniques cover a significant range of assumptions – from simple, linear boundaries to complex.

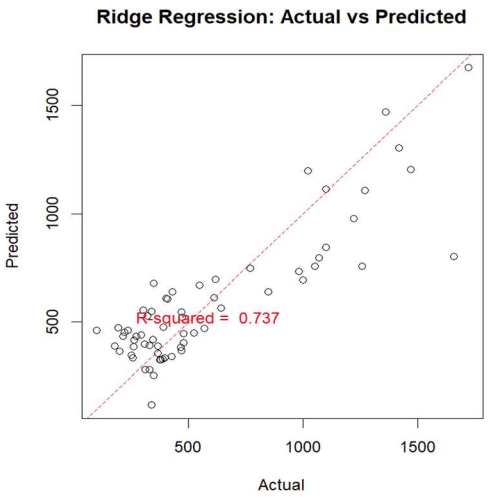
Plot of Actual vs Predicted value.

For Property 01:

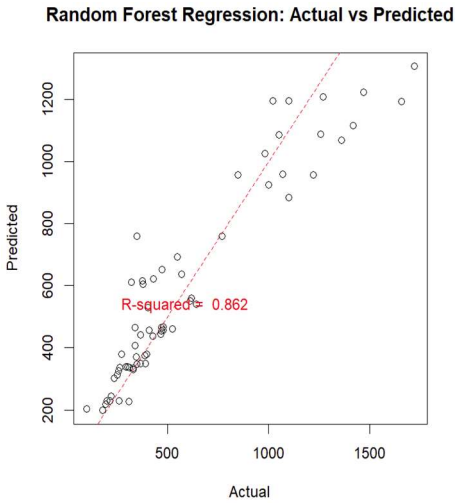
MLR



Ridge regression

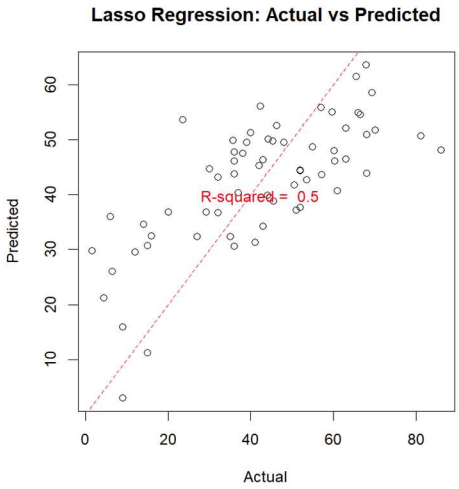


Random Forest

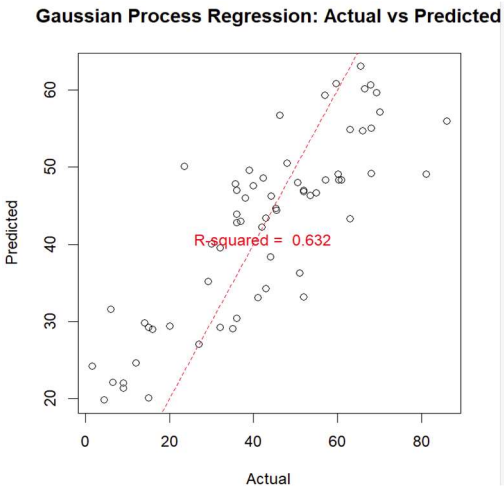


For Property 02:

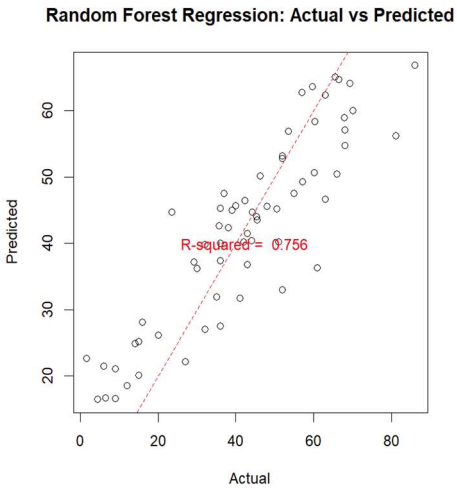
Lasso regression



GPR



Random Forest

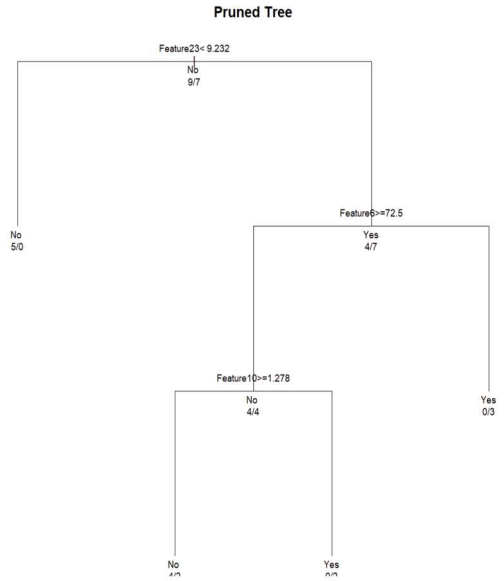


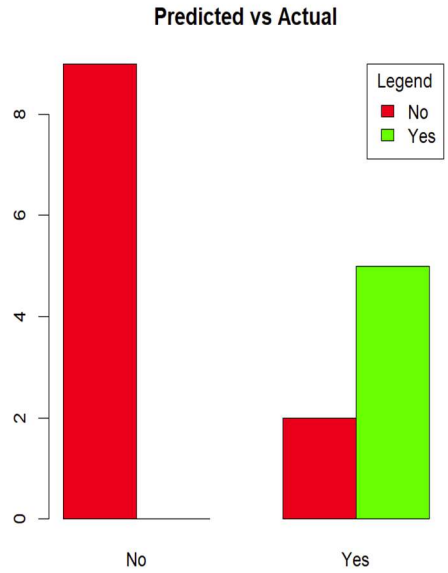
For property 03:

Logistic Regression

```
> (misclass <-  table(glm.pred_train, t
                truth
glm.pred_train  0  1
             0 24 13
             1  7 20
> #Apply model and repeat on training
> glm.probs_test <- predict(glm.fit,te
Warning message:
In predict.lm(object, newdata, se.fit,
  prediction from rank-deficient fit;
> glm.pred_test <- ifelse(glm.probs_te
> (misclass <- table(glm.pred_test, tr
                truth
glm.pred_test 0 1
             0 5 4
             1 4 3
```

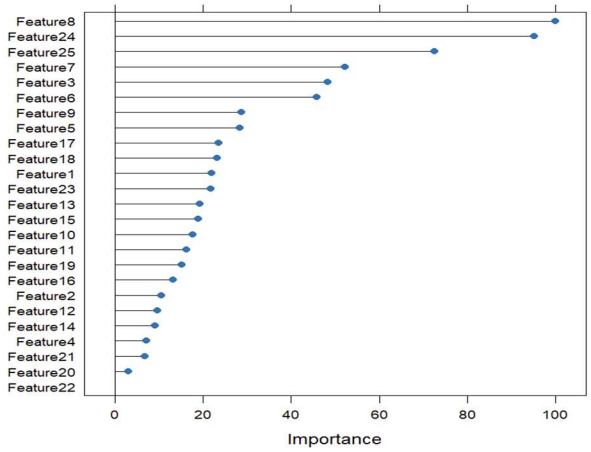
Decision tree






```
             High
tree.pred_train No Yes
          No  20  22
          Yes 11  11
> tree.pred_test=predict(pfit,test_dt,type="class")
> with(test_dt,table(tree.pred_test,High))
             High
tree.pred_test No Yes
          No   9   2
          Yes  0   5
```

Random forest




```
             High
tree.pred_train No Yes
          No  30   3
          Yes  1  30
> tree.pred_test=predict(train.rf,test_dt)
> with(test_dt,table(tree.pred_test,High))
             High
tree.pred_test No Yes
          No   5   3
          Yes  4   4
```

<u>Missing Data Prediction:</u>

Property 01 and Property 2: best model = Random forest

Property 03: best model = Logistic Regression

Property 1 prediction	Property 2 prediction
820.6748	51.9419
822.8249	39.03536
815.8352	48.91036
746.8991	54.09445
724.353	53.80829
675.131	60.66492
718.2627	44.27898
524.2582	51.03178
501.8421	56.38314
510.5049	60.24789

```
> # Check the predictions
> print(missing_data$glm_pred_class)
 [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
>
```

<u>Observation:</u> Based on the input and model evaluation metrics, none of the 10 systems meet the defined criteria for all three properties simultaneously. Based on the prediction from all three models for each property 1 and 2, the one with the large values were chosen. While models for property 1 and 2 predict large values, the models for property 3 consistently predict a class of 0, indicating that the systems don't exist.

Possible reason for the observation: Lack of data might be the cause for the model being unable to recognize the underlying pattern of the data. Or maybe the data in the first 10 systems may contain noise.

Answer(i) I feel confident with my predictions based on the comprehensive set of techniques I have chosen for predicting all three property values. However, slightly better accuracy and evaluation metrics for property 2 and 3 respectively would have tremendously boosted the confidence.

My confidence would have increased if I had data for more systems for following reasons:

a) Enhanced Model robustness: more data aids in reducing the variability of model predictions.
b) Training diversity: More rows can capture wider range of data, leading to models performing good across diverse cases.
c) Better comprehension of underlying patterns: with more systems, complex relationships and patterns can be spotted and leveraged for prediction.

Answer (ii) Following are the ways to assess the uncertainty in the models:
1. Confidence intervals: for regression coefficients in the models such as Lasso regression, we can calculate confidence intervals yielding a range where the true coefficient is likely to exist with a certain probability.
2. Cross validation: perform cross validation to estimate the performance of model on unseen data. The variation in performance metrics across folds can give you insight into the model's stability.
3. Model-specific uncertainty: for GPR, the model itself provides a measure of uncertainty for its predictions.

Answer (iii) To check the unbalance in the dataset I filtered data for 0 and 1 cases for Property 3 column. 40 systems had value as '1' or 'Yes' and rest 40 as '0' or 'No'. Hence there was no issue with the unbalanced dataset.

If in case we face the issue of unbalanced dataset then following are the approaches to handle:
1. Random sampling: We randomly remove some majority class samples to obtain the balance. Additionally, we can run multiple times to make sure not specific to random selection.
2. System Minority Oversampling Technique (SMOTE): it enhanced the representation of the minority class by identifying
ithin the feature space, constructing a line connecting these instances, and generating new samples along
ess of oversampling aids in balancing the class distribution.

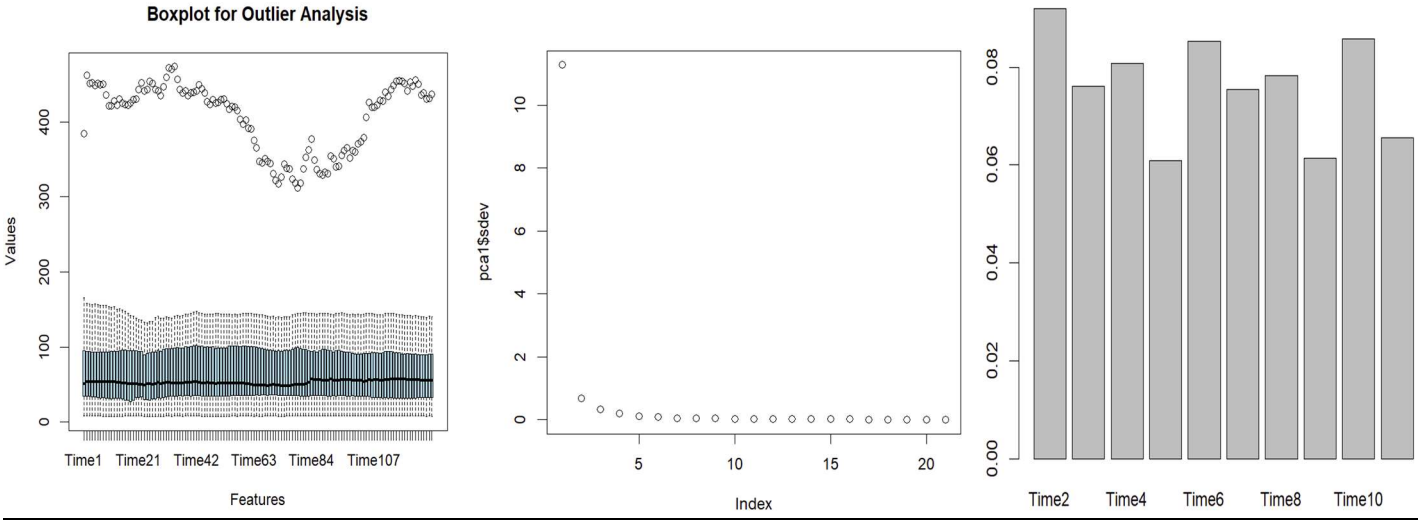
Problem 02:

Step 01: I removed the missing column from the main dataset. Now, we are left with company 5 – 25 and corresponding value of stock from time 0 to 128. For training and testing, we have considered those columns that does not contain "?"

EDA (Exploratory Data Analysis) – Steps 02 to 04 are common to all the three regression models used.

Step 02: Scaling of the data: Due to the significant variation in the magnitude of the data for each stock value, I performed scaling.

Step 03: Outlier Analysis: plotted boxplot to observe the outliers. Outliers were removed based on the 3 IQR range criteria.



Step 04: Principal Component Analysis: Using PCA reduces the dimensionality of the dataset by condensing features into fewer principal components.

PCA Elbow Plot: We can observe that approximately 2 principal components give us 90% variance.

PCA VIP PLOT: this plot helped me to identify the features with higher importance for the analysis. Based on the height of the bar, Feature 2, 3, 4, 5,7,8, 9, 10 & 11 are the important features or Time values

The PC1 component consists of Features 2 – 11
The PC2 component consists of features 8, 19, 27, 38, 53, 77, 69, 72

Step 06: I split the training to testing data in the ratio of 80:20 for understanding accuracy of the training and testing data.
Step 07: Respective model based on the property type was built based on the type and parameters required.
Step 08: Calculated RMSE of training and testing data and R-squared value.
Step 09: As the given dataset is continues, we cannot remove any descriptor here.

Answer(i)

Model	Train RMSE	Test RMSE	R-squared
Lasso regression	18.17616	43.57231	0.4997178
PCR	15.16228	41.31031	0.6317625
Random forest	22.905548	48.48654	0.7557386

Predicted property	Lasso regression	PCR	Random forest
Company 1	41.4135	61.3048	51.69898
Company 2	61.4665	87.7221	98.4075
Company 3	55.7682	55.2096	66.0292
Company 4	66.9636	68.2637	74.8099

I chose following models for the below reasons:
Lasso regression is chosen for its ability to perform feature selection, reducing potential overfitting of the data by penalizing less significant features.
PCR is used to address multi-collinearity by transforming the predictors into a set of a set of orthogonal components, that is the PC components.
Random Forest is chosen for its robustness to outliers and its ability to handle non-linear or complex relationships.

Justification for selection of features:

In case of PCR: dimensionality reduction was achieved via PCA and two principle components contributed to variance of 90%. PC 1 & PC 2 consisted a total of 19 features.

```
In data(time_only) : data set 'time_only' not found
> time_only_std<-as.data.frame(scale(time_only))
> pca1<-prcomp(time_only_std)
> pca1$sdev/sum(pca1$sdev)
 [1] 8.708409e-01 5.159530e-02 2.545137e-02 1.540544e-02 7.781891e-03 6.533817e-03 3.603117e-03 3.323873e-03 2.652075e-03
[10] 2.342080e-03 2.007154e-03 1.688309e-03 1.669146e-03 1.298655e-03 1.075183e-03 9.530181e-04 6.505580e-04 5.295251e-04
[19] 3.254172e-04 2.732049e-04 2.362803e-17
> loads<-pca1$rotation
> scores<-pca1$x
> #Select number of PCs
> plot(pca1$sdev)
> (pca1$sdev[1]+pca1$sdev[2]+pca1$sdev[3]+pca1$sdev[4])/sum(pca1$sdev)
[1] 0.963293
>
> #VIP Calculation
> #update loadings w/ reduced no. of PCs
> loads_vip<-loads[,1:4]
> property_vip<-loads_vip[1,]
> features_vip<-loads_vip[2:11,]
> weight_vip<-property_vip*features_vip
```

In case of Lasso regression: feature selection was done by applying a penalty to the absolute size of regression coefficients and effectively shirnking to zero. Hence, overall 57 features were utilized.

The regulization parameter of lambda was set to zero.

```
descriptors_train_ridge<-train_ridge[,! names(train_ridge) %in% c("Property2")]
descriptors_test_ridge<-test_ridge[,! names(test_ridge) %in% c("Property2")]
descriptors_train_ridge<-as.matrix(descriptors_train_ridge)
descriptors_test_ridge<-as.matrix(descriptors_test_ridge)
mdl_ridge<-glmnet(descriptors_train_ridge,train_ridge$Property2,alpha=1)
mdl_ridge_cv<-cv.glmnet(descriptors_train_ridge,train_ridge$Property2,alpha=1)
best_lambda<-mdl_ridge_cv$lambda.min
mdl_ridge_best<-glmnet(descriptors_train_ridge,train_ridge$Property2,alpha=1,lambda=best_lambda)
coef(mdl_ridge_best)
pred_train_ridge<-predict(mdl_ridge,s=best_lambda,newx=descriptors_train_ridge)
pred_test_ridge<-predict(mdl_ridge,s=best_lambda,newx=descriptors_test_ridge)
```

Random forest build multiple decision trees and merges them together. Each feature importance are calculated and then the decision tree is build.

Answer(ii)

The confidence in each model can be described as follows:
Lasso regression: With a moderate RMSE and the least R2 value among the three models, confidence in the setting might be lower as compared to Random Forest. However, Lasso's strength in feature selection can make it robust, if the dataset has irrelevant features.

PCR: The PCR has relatively high Test RMSE value, suggesting less accuracy. The R2 value is moderate, indicating a fair fit of the model to the data. PCR can be sensitive to the number of PC components chosen, which might affect the model's performance.

Random Forest: This model shows the lowest RMSE values and highest R2 value. This model are generally less sensitive to the parameters and can handle a wide range of data types as well.

Overall, random forest seems to offer the most reliable predictions and is less sensitive to parameter changes, suggesting good generalizability across different datasets.