Statistical learning applied on graduate admission dataset

Silviu Filote 1059252,
Jonathan Bommarito 1068755

June 2023

Contents

1 Introduction

The statistical learning project aims to apply some approaches described in the "An Introduction to Statistical Learning"[1] to a chosen dataset and select the best model which can explain better our response variable. The main goal is to find a model which can be accurate and at the same time interpretable, that means the complexity will be reduced in order to provide an understandable solution for the case study.

In the following sections will be introduced the dataset adopted and all the approaches taken into consideration.

2 Dataset

The selected dataset is called "Graduate Admission 2" and it's available on kaggle site [2]. The dataset includes 500 rows without missing values and it is composed by the following variables:

Continuous variables:

- Graduate Record Examinations Scores (GRE): test that measures abstract thinking in areas such as reading comprehension, writing, and mathematics;
- Test Of English as a Foreign Language (TOEFL): it's the score of english language test;
- Cumulative Grade Point Average (CGPA): is based on all coursework completed for bachelor's degree, representing overall average;
- Chance of Admit (Admit): percentages that indicates the probability of being admitted.

Discrete variables:

- University Rating (UniRatings): It refers to the applicant's undergraduate university. It would be fair to say that not all universities carry the same repute around the world. There are a few universities that are recognised better, it is a parameter that may play a role in deciding which student is to be offered admission when two similar profiles pop up;
- Statement of purpose (SOP): is a short essay that highlights the educational background, achievements, and goals (it also incorporates values at half levels);
- Common Letter of Recommendation (LOR): score from 1 to 5 (with half values) assigned to the letter of recommendation;
- Research Experience (Research): boolean that indicates any professional or academic research activity acquired in any research field in the public or private sector.

In the following sections, we will try to explain the Admit variable using all the remaining variables described above as regressors, treating the discrete variables as dummy variables. We will start implementing linear models in order to observe how simple models behave and then compare them with more complex models such as generalized additive models (GAMs) and tree models. At the end, we will select the best model which better explains the response variable, taking into account the complexity of this one.

The following images show the distribution of the previously described variables and how they are correlated with each other:

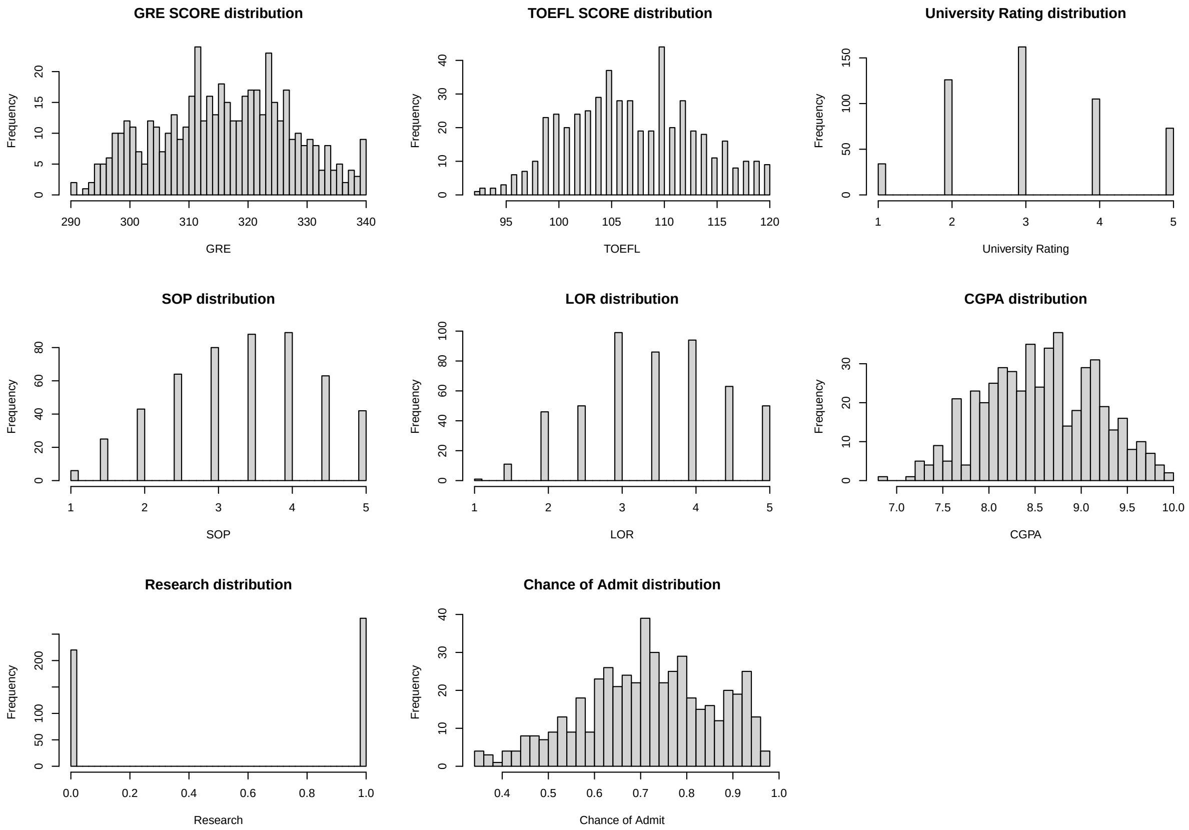
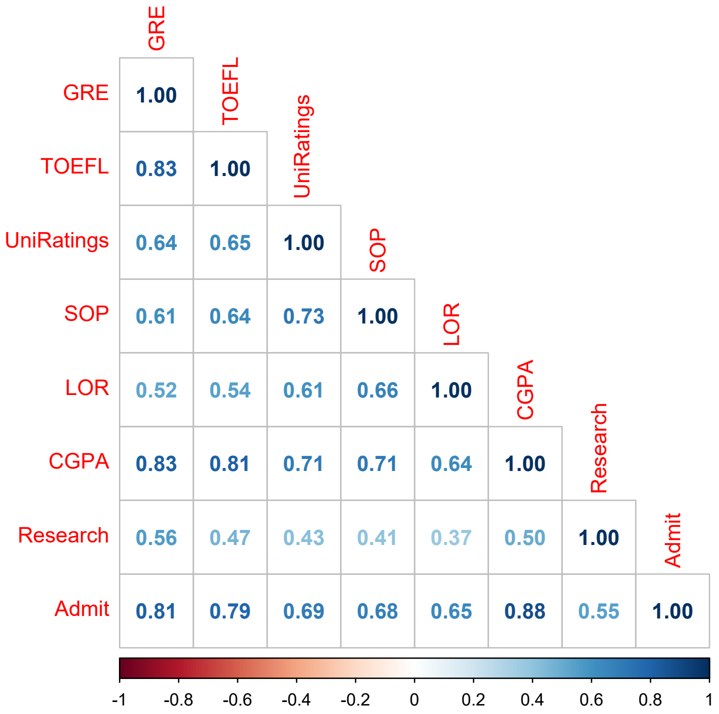


Figure 1: Empirical distribution of all the variables inside the dataset



	GRE	TOEFL	UniRatings	SOP	LOR	CGPA	Research	Admit
GRE	1.00							
TOEFL	0.83	1.00						
UniRatings	0.64	0.65	1.00					
SOP	0.61	0.64	0.73	1.00				
LOR	0.52	0.54	0.61	0.66	1.00			
CGPA	0.83	0.81	0.71	0.71	0.64	1.00		
Research	0.56	0.47	0.43	0.41	0.37	0.50	1.00	
Admit	0.81	0.79	0.69	0.68	0.65	0.88	0.55	1.00

Figure 2: Correlation matrix between all the variables inside the dataset

From the correlation matrix we can observe that all the variables are positively correlated with the response variable and research is the regressor which has the lowest correlation with the response variable. Furthermore, from the Figure 2 is possible to observe that the continuous regressors have higher linear correlation than the discrete ones.

2.1 Outliers

Using an outliers' detection tool in R it's possible to see a few outliers in the dataset.

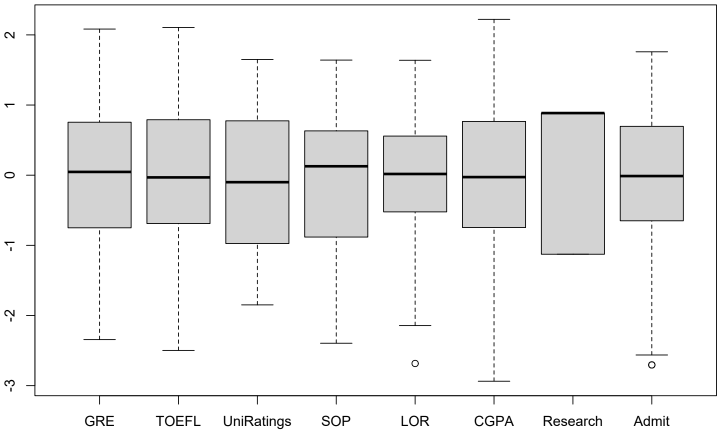


Figure 3: Detecting possible outliers in the regressors' distribution using boxplot

In order to prevent failures and mistakes, all the outliers are going to be removed from the dataset using IQR method.

3 Linear regression methods

In this chapter we will use two methods to establish the goodness of the model that are respectively validation and cross-validation. For the validation we will split our dataset in two parts where the 70% percent will be used to train the model and the remaining 30% to validate it. Once created the training dataset we fitted the model using all regressors and then removing at each iteration the least significant. The fitted model is the following where data is the complete dataset and train is an index containing the 70% of the index into the dataset:

```
lm_model <-  lm(formula = Admit ∼ GRE + TOEFL + CGPA + LOR + Research,
                          data = data, subset = train)
```

Table 1: Result of applying the linear model in order to explain the response variable using the most important regressors.

Coefficients	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.257	0.113	-11.085	<2e-16	***
GRE	0.002	0.001	2.860	0.0045	**
TOEFL	0.002	0.001	2.639	0.0087	**
CGPA	0.129	0.011	11.918	<2e-16	***
LOR2	0.053	0.027	1.944	0.0527	.
LOR2.5	0.061	0.027	2.235	0.0260	*
LOR3	0.061	0.026	2.306	0.0217	*
LOR3.5	0.079	0.027	2.964	0.0033	**
LOR4	0.086	0.027	3.228	0.0014	**
LOR4.5	0.093	0.027	3.407	0.0007	***
LOR5	0.109	0.028	3.882	0.0001	***
Research1	0.029	0.007	3.973	8.71e-05	***

We kept into the model the most important regressors and we removed: UniRatings and SOP. Subsequently, as the model responses is a percentage value, we checked our model predicts values between 0 and 1 in order to ensure it didn't estimate anomalous values and from the check no problems were encountered. Furthermore, we analyzed the trained model residuals.

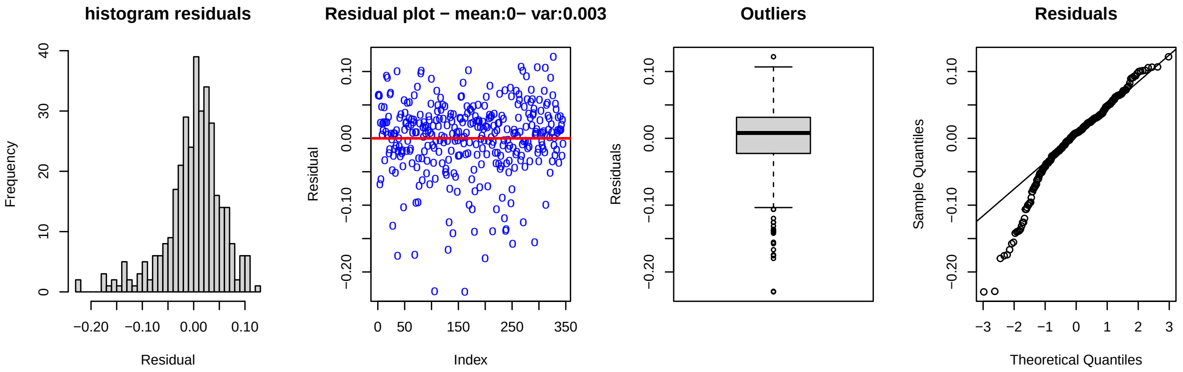


Figure 4: Residuals plotting made on the linear model

The residuals plotting highlights the following features:

- `qqnorm` plot and histogram show a negative skewness;
- There are many outliers;
- The Shapiro-Wilk test with a p-value $= 3.399e^{-11}$ confirms that the distribution of the residuals is non-normal;
- Studentized Breusch-Pagan test with a p-value $= 0.001746$ confirms that the distribution of the residuals is heteroscedastic.

In order to verify the model is reliable, we also implemented bootstrap. Since the residuals are heteroscedastic and non normal we can't trust the coefficients estimated using the linear approach. The bootstrap purpose is to check if the coefficients are well estimated as well as the standard deviation without bothering the normality assumption of the residuals.

```
boot <- (data = as.data.frame(data[train,]), statistic = get_model, R = 1000)
```

Table 2: Boostrap test applied on the training dataset in order to estimate the coefficients of a linear model

Coefficients	original	std. error
(Intercept)	-1.257	0.131
GRE	0.002	0.001
TOEFL	0.002	0.001
CGPA	0.129	0.011
LOR2	0.053	0.026
LOR2.5	0.061	0.027
LOR3	0.061	0.025
LOR3.5	0.079	0.026
LOR4	0.086	0.025
LOR4.5	0.093	0.026
LOR5	0.109	0.024
Research1	0.029	0.009

As shown in Table 2 the coefficients and standard error are similar to those in the fitted model, it means we can trust our estimates even if there is variability left in the residuals that our model doesn't capture.

3.1 Subset selection methods

In this section we fit again the linear regression model using all regressors in the dataset, adding log, exponential, square and polynomial with 3 degree of freedom, for each continuous variable. This time we implemented the backward stepwise algorithm in R in order to let the algorithm decide which regressors include in the model, choosing by best AIC value. The model selected by the algorithm is the following:

```
stepwise_model <-  lm(formula = Admit ~ UniRatings + LOR + Research + exp(CGPA) +
                        log10(GRE) + log10(TOEFL) + log10(CGPA) + poly(CGPA, 3),
                        data = data, subset = train)
```

Remarks:

- The coefficients estimated from the model are 42 and the algorithm selected only 20 which minimize the AIC;
- The stepwise algorithm included all the transformation for the CGPA regressor and for the other continuous variables only the log_{10} transformation;
- Total regressors selected are 10;
- Stepwise algorithm also includes UniRatings regressor in the model;
- Residual standard error is lower and adjusted R-squared is higher than the normal `fit_lm`;
- Residuals tests confirm again heteroscedasticity and non-normal distribution;
- The predictions of the model are again reliable and not out of bounds.

3.2 Lasso

In this final section we fitted the model using all regressors like we did for backward stepwise algorithm including the previously transformations too. Firstly we divided our dataset in two parts that are respectively training and test sets. After that, we executed Lasso algorithm in order to reduce the model complexity and cross-validation method to find the optimal lambda value.

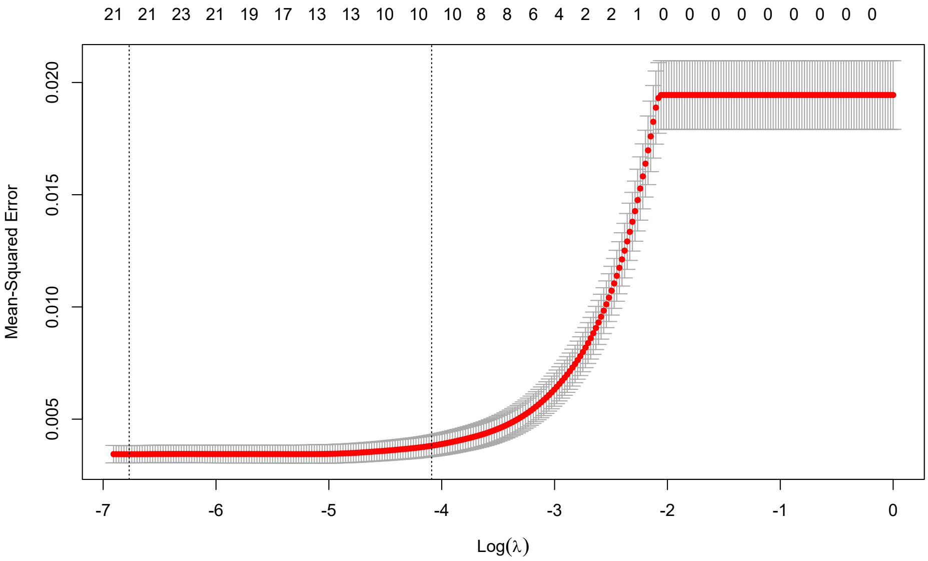


Figure 5: Cross-validation method applied on the training dataset for the purpose of choosing the best penalization (lambda) that reduces the model complexity. The two vertical lines highlight two possible values for lambda

As the above plot shows there are highlighted by vertical lines two different values for the λ parameter. The left-most line indicates lambda value which minimize MSE with a lower penalization (R calls it `lambda.min` $\lambda_{min} = 0.001$), while the second value reduces the model complexity using a higher penalization including only 10 regressors (R calls it `lambda.1se` $\lambda_{min} = 0.017$). For our purposes we have decided to choose the second value of lambda λ_{min} which minimize drastically the model complexity without losing too much interpretability.

The final model estimated from the lasso algorithm using λ_{min} includes β_0 plus 10 regressors.

```
lasso_model = Admit ∼ sGRE + cGRE + TOEFL + lTOEFL + rTOEFL + CGPA + lCGPA + sCGPA +
                              rCGPA + Research1
```

s = squared - c = cubic - r = root

The residual analysis confirms that:

- we have a prominent negative skewness;
- residuals are heteroscedastic and not normal distributed.

3.3 Model improvements

In this section we have applied the *arcsin* transformation to the response variable Admit in order to reduce the variability of the distribution of the residuals. We can execute this operation because the response variable is positive and the entire range of values are included in the *arcsin* domain.
Regardless of the results, this transformation involves an increase in the complexity of the model and this is not the goal of our analysis, also the RMSE test of this model is the worst compared to the other ones.

```
lm_imp_model <- lm(formula = asin(sqrt(Admit)) ∼ GRE + TOEFL + CGPA + LOR + Research,
                              data = data, subset = train)
```

The residual analysis clarify that:

- Breusch-Pagan test unlike before confirms that the residual distribution is now homoscedastic;
- the Shapiro-Wilk normality test reveals once again that the residuals are not normally distributed.

3.4 Final considerations about linear models

In conclusion we can summarize the models implemented in the following table:

Table 3: All the linear regression models implemented during this section with the most important statistics

Model	RMSE test	CV RMSE	Shapiro-Wilk train	Breusch-Pagan train	Adj. R^2 train
`lm_model`	0.064	0.059	pv = 3.345e-11	pv = 0.002	0.837
`stepwise_model`	0.064	0.059	pv = 3.999e-11	pv = 0.008	0.841
`lasso_model`	0.066	0.060	pv = 1.628e-08		0.809
`lm_imp_model`	0.140	0.067	pv = 1.004e-08	pv = 0.095	0.857

As shown in the previous table, as far as the RMSE test value and CV RMSE value are concerned, the first three models are similar. Unfortunately, the residuals of these models are heteroscedastic, as Breusch-Pagan test confirms. The `lm_imp_model`, instead, has homoscedastic residuals but its RMSE test and CV RMSE are higher than the other models. Furthermore, as described previously, this model uses an “arcsine” function which cause a complexity increment. In conclusion, we selected `lm_model` as the best linear model because it is surely the most interpretable one and it performs well in validation, although its residuals are heteroscedastic.
We will use this model to do other comparisons with more complex models that will be discussed in the following sections.

4 Generalized additive models (GAMs)

In the previous section we implemented linear regressor model to explain "Admit" variable assuming linear relations between it and the regressors. In this section we will fit generalized additive models in order to catch also non-linear relations through the use of splines. The following image shows a summary of a model fitted using all regressors and wrapping continuous variables with `s()` command in order to use smooth splines.

4.1 GAM with all regressors

In this first non linear-model we didn't impose degrees of freedom for splines, leaving R the choice.

```
gam_all_regressors <-  gam(formula = Admit ∼ s(TOEFL) + s(CGPA) +  s(GRE) + Research +
                           SOP + LOR + UniRatings, data = data, subset = train)
```

Table 4: GAM with all regressors

	Df	Sum Sq	Mean Sq	F value	Pr(>|f|)	
s(TOEFL)	1	4.272	4.272	1375.793	<2.2e-16	***
s(CGPA)	1	1.166	1.166	375.439	<2.2e-16	***
s(GRE)	1	0.045	0.045	14.634	0.0002	***
Research	1	0.055	0.055	17.606	3.547e-05	***
SOP	8	0.062	0.008	2.507	0.0118	*
LOR	7	0.049	0.007	2.248	0.0304	*
UniRatings	4	0.027	0.007	2.193	0.0696	.
Residuals	312	0.969	0.003			

Table 5: non-linear splines component in GAM with all regressors

	Df	Npar F	Pr(F)	
s(TOEFL)	3	1.339	0.262	
s(CGPA)	3	2.778	0.041	*
s(GRE)	3	1.109	0.346	

The previous tables demonstrate how R split each spline into two component, evaluating them separately. The first one is the linear component, which turned out to be significant in the model. The second one, instead, is the non-linear component where, for each spline, R imposed three degrees of freedom. This second component, unlike the previous one, isn't significant for the model.
In the figure 6 are shown the relations between each regressor and the response variable:

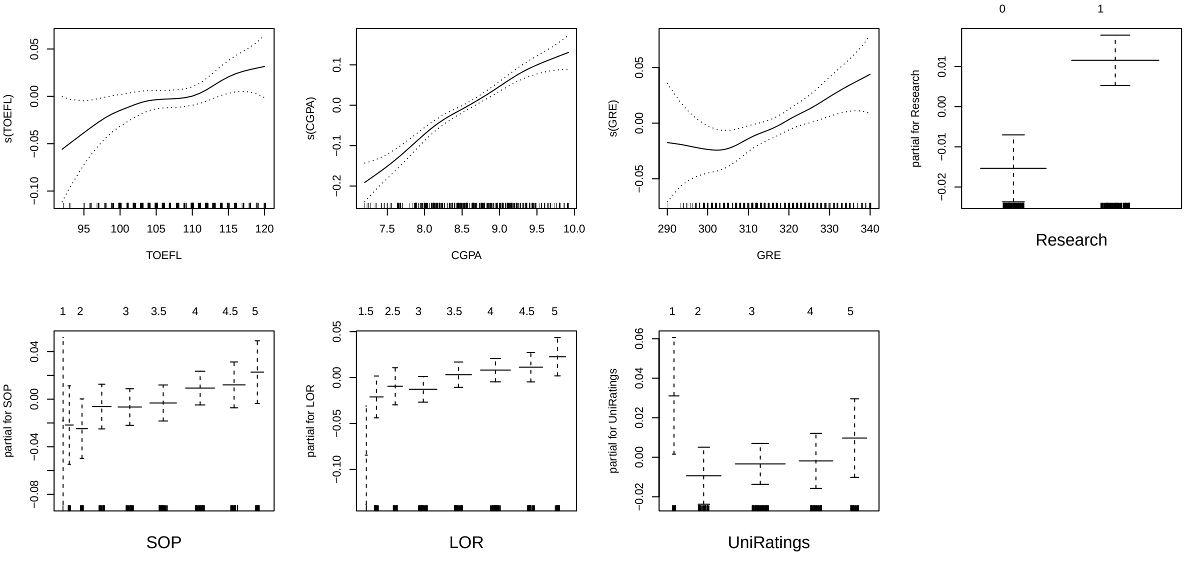


Figure 6: GAM with all regressors

By the previous figure, it is possible to note that also splines show a slightly wavy linear trend, which confirm what we said previously that is the significant component in the spline is the linear one.

4.2 GAM with only significant regressors

We continue removing non-significant regressors, letting R choice the degrees of freedom for the splines once again.

```
gam_sigreg <- gam(formula = Admit ~ s(TOEFL) + s(CGPA) + s(GRE) + Research,
                  data = data, subset = train)
```

Table 6: GAM with only significant regressors

	Df	Sum Sq	Mean Sq	F value	Pr(>|f|)	
s(TOEFL)	1	4.230	4.230	1276.445	<2.2e-16	***
s(CGPA)	1	1.205	1.205	363.609	<2.2e-16	***
s(GRE)	1	0.040	0.040	11.920	6.275e-04	***
Research	1	0.054	0.054	16.334	6.607e-05	***
Residuals	331	1.097	0.003			

Table 7: non-linear splines component in GAM with only significant regressors

	Df	Npar F	Pr(F)	
s(TOEFL)	3	1.406	0.241	
s(CGPA)	3	2.170	0.091	.
s(GRE)	3	0.931	0.426	

As happened previously, non-linear components in the splines aren't important. Now we have removed non-significant regressors, we want to figure it out if we can improve `gam_sigreg` changing splines degrees of freedom.

4.3 GAM with the most performing degrees of freedom

At this point we wondered if there was a combination of degrees of freedom for spline which perform better the other models. Up to know we saw, even if we used spline, the trend for continuous variable is similar than we have with linear regression, so we want to verify that the best choice for spline is to choose just one degree of freedom. To do this, we implemented an algorithm which fit models trying each degree of freedom combination, starting from 1 until 5, computing for each model RMSE value in training and validation datasets.

Each model estimated by the algorithm turned out to have similar RMSE value in validation. The lowest value, indeed, is 0.06397 while the highest is 0.06480. As the algorithm simply looking for the model with the minimal RMSE test value we need to verify the selected model isn't too complex which wouldn't be justified, as the differences compared to all the others, in terms of RMSE value is low. The following table describes the selected model by the algorithm:

```
gam_suggested <- gam(formula = Admit ~ TOEFL + s(CGPA, 5) + GRE + Research, data =
                     data, subset = train)
```

Table 8: GAM suggested by the algorithm

	Df	Sum Sq	Mean Sq	F value	Pr(>|f|)	
TOEFL	1	4.212	4.212	1278.946	<2.2e-16	***
s(CGPA, 5)	1	1.227	1.227	372.395	<2.2e-16	***
GRE	1	0.041	0.041	12.519	4.595e-4	***
Research	1	0.050	0.050	15.126	1.212e-4	***
Residuals	336	1.107	0.003			

Table 9: non-linear splines component in GAM suggested by the algorithm

	Df	Npar F	Pr(>|f|)	
s(CGPA, 5)	4	2.3813	0.0514	.

The suggested model use 1 df[1] for TOEFL and GRE splines but 5 df of freedom for CGPA.

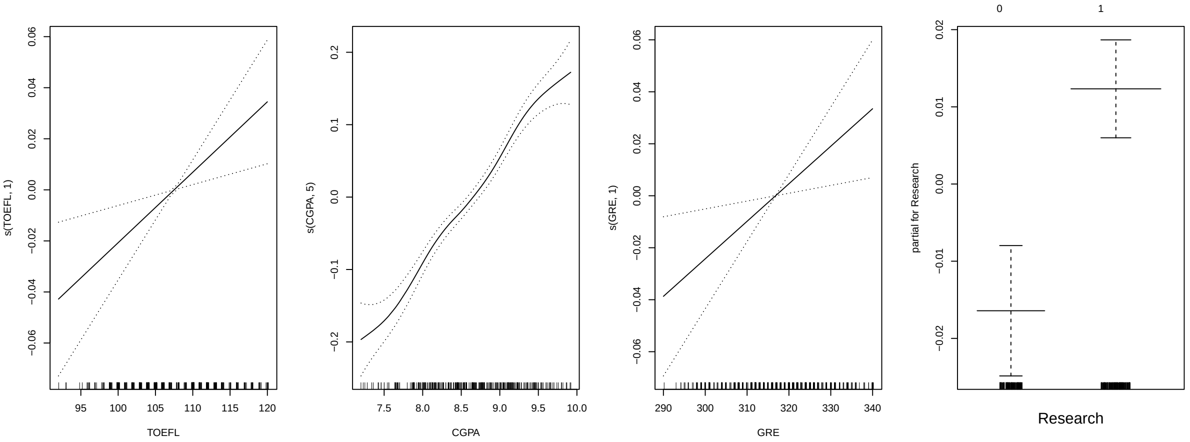


Figure 7: plot regressors for the suggested model

As we can see, even if we use 5 df for CGPA spline, it keeps a trend similar to the linear one. In the following table are shown the differences between each fitted model:

Model	RMSE train	RMSE test	Shapiro-Wilk train	Breusch-Pagan train
`lm_model`	0.05531	0.06425	pv = 3.345e-11	pv = 0.0019
`gam_all_regressors`	0.05299	0.06628	pv = 1.614e-11	pv = 0.0156
`gam_sigreg`	0.05639	0.06443	pv = 1.519e-11	pv = 0.0017
`gam_suggested`	0.05663	0.06397	pv = 9.060e-12	pv = 0.0017

In all the Generalized Additive models, we have similar RMSE test value. Obviously the model suggested by the algorithm has the lowest value, otherwise it wouldn't have been chosen as the best one. Nevertheless, that model includes a regressor with an excessively high complexity. For this reason, using RMSE test and complexity as criteria, we can assert the best GAMs is `gam_suggested` that includes exclusively significant regressors and at the same time it has the lowest RMSE value. Although `gam_suggested` is the best generalized additive model, we can observe its RMSE test value is still higher than `lm_model` value. So as `lm_model` has the lowest value and it doesn't use spline, helping the model interpretation, we still consider `lm_model` as the best model analyzed up to now.

[1] degree of freedom

5 Regression Trees

In this section we are going to apply trees to our dataset and see how well they perform compared to the other models. In the first place we will build a simple regression tree and apply the pruning method in order to reduce complexity and overfitting. The penalization factor that involves pruning is chosen using a cross-validation approach.

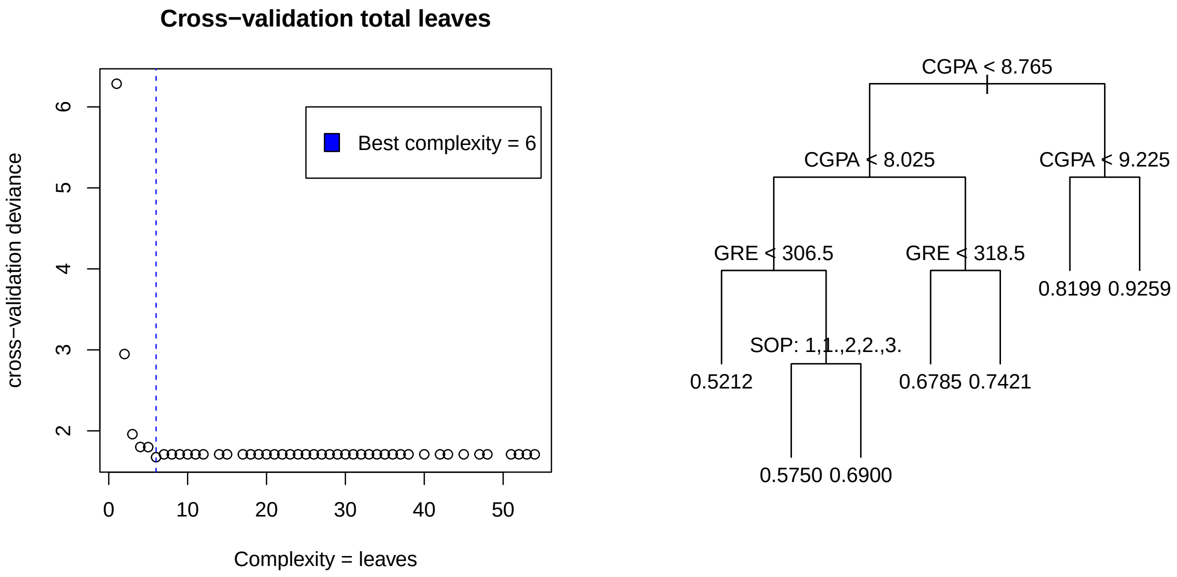


Figure 8: Left: The cross-validation chooses the number of total leaves of the tree, that indicate the complexity of the tree and where to stop the splitting - Right: Tree pruned model using the best complexity parameter discovered in cross-validation

We have applied the cross-validation in order to get the best number of total leaves, which indicates the complexity of the model. The output of the cross-validation algorithm is 6, but we are going to take 7 because it is a more stable parameter and after that the deviance stabilizes. We then implemented the pruned tree using the total number of leaves equal to 7 and the result is displayed in Fig. 8.

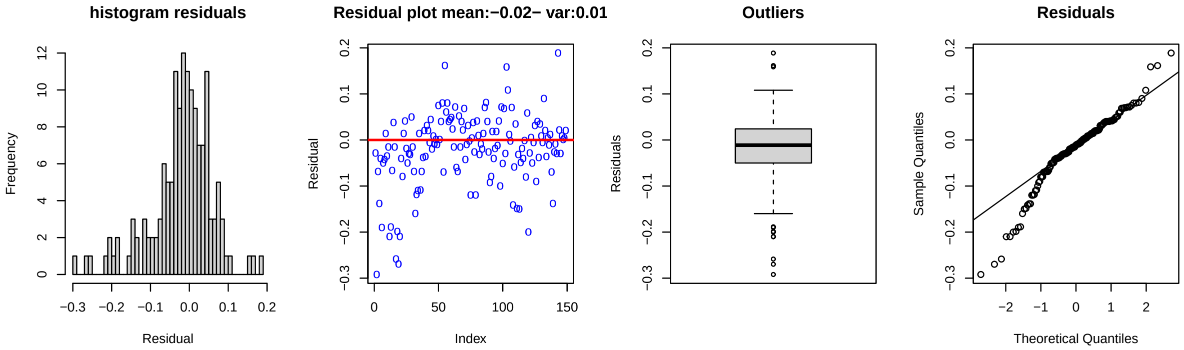


Figure 9: Residuals obtained in the validation/prediction phase from tree model after pruning

The residual analysis highlights that:

- Shapiro-Wilk normality test denotes that residuals are not normally distributed;
- There is also an evident heteroscedasticity inside the residuals' distribution.

Trees are very easy to understand and explain but generally they are non-robust and contains high variance, for this reason we are going to consider more advanced algorithms base on trees and examine their behaviour.

5.1 Bagging

Bagging, also known as Bootstrap aggregating, is an ensemble learning technique that fits many training data as trees using bootstrap in order to create new training datasets. Once the trees are created for each dataset the final prediction is made taking into account all the built trees. Bagging models do not prune the tree in the training phase, for this reason trees tend to have really low bias but high variance. The variance is reduced by creating numerous trees during the prediction phase. Bagging models performs the splits of each tree considering all the regressors in the dataset, therefore it's really important to carefully choose the right number of trees we want to build in the training phase to not achieve any overfit.

In this section we are going to implement bagging models on our regressors to explain "Admit", selecting the best number of trees using cross-validation.

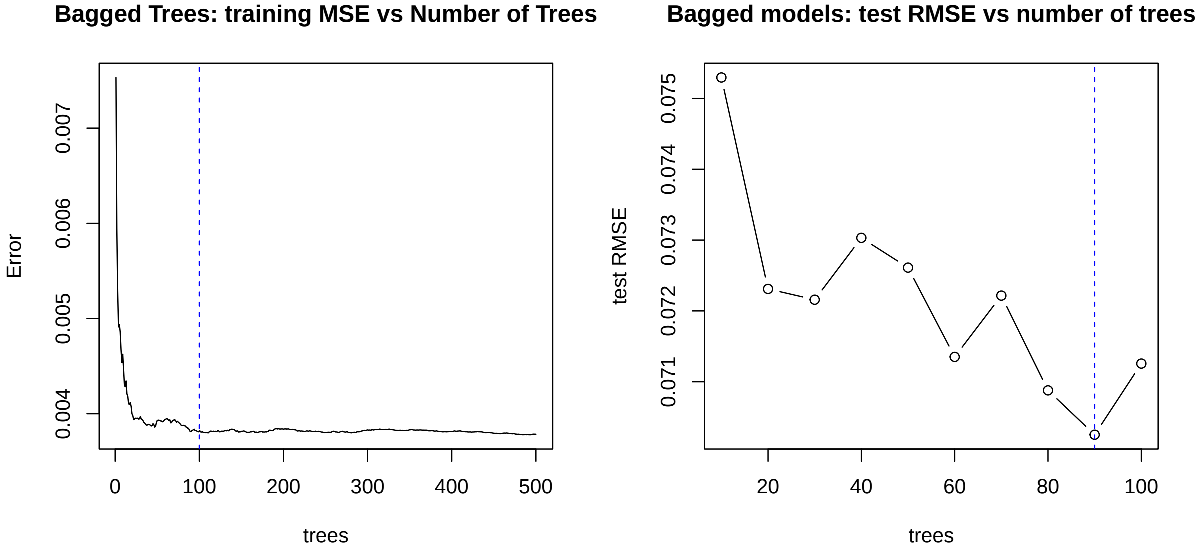


Figure 10: Left: The behaviour of a bagging model using a high number of trees compared to the training MSE. - Right: We fitted 10 different bagging models changing `ntree` parameter from 10 to 100 with a span of 10 and then calculating for each iteration the test RMSE.

There is a clearly overfitting if we consider a number of tree higher than 100, so we have implemented 10 different models with a number of trees under 100 choosing the one that optimize the test RMSE. From Fig. 10 we can see that the best model in optimization is the one with `trees = 90`.

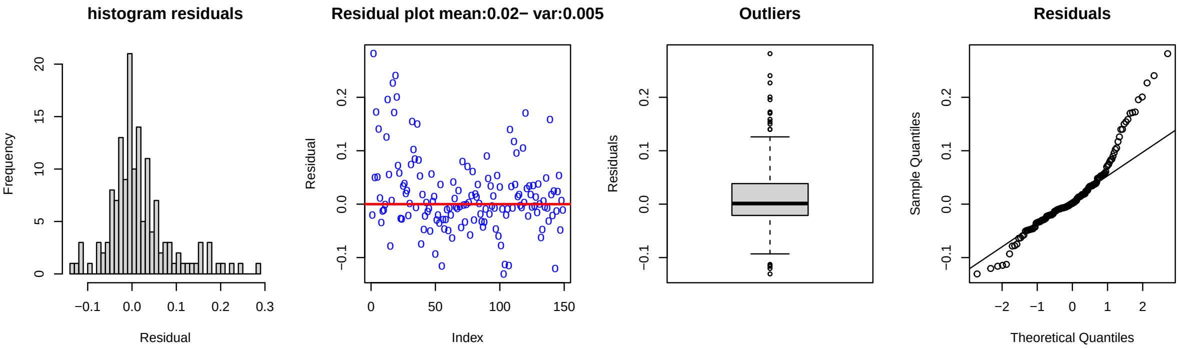


Figure 11: Residuals obtained in the validation/prediction phase from the optimized bagging model

The residual analysis highlights that:

- the residuals are not normally distributed and they are still heteroscedastic;
- there is a noticeable positive skewness and this is the first time we have observed it in an algorithm.

5.2 Random forests

The random forest method works in the same way as the bagging model, but splits are constrained to a subset of the total regressors in the dataset. This constraint approach is made in order to reduce the variance by considering different builds of trees because at each split we use different regressors.
Random forest method compared to Bagging method has a lower learning rate and for this reason random forest approach needs a higher number of trees.

We are going to choose the optimal number of trees using cross-validation and change the subset of the regressors used at each split to see how the models respond.

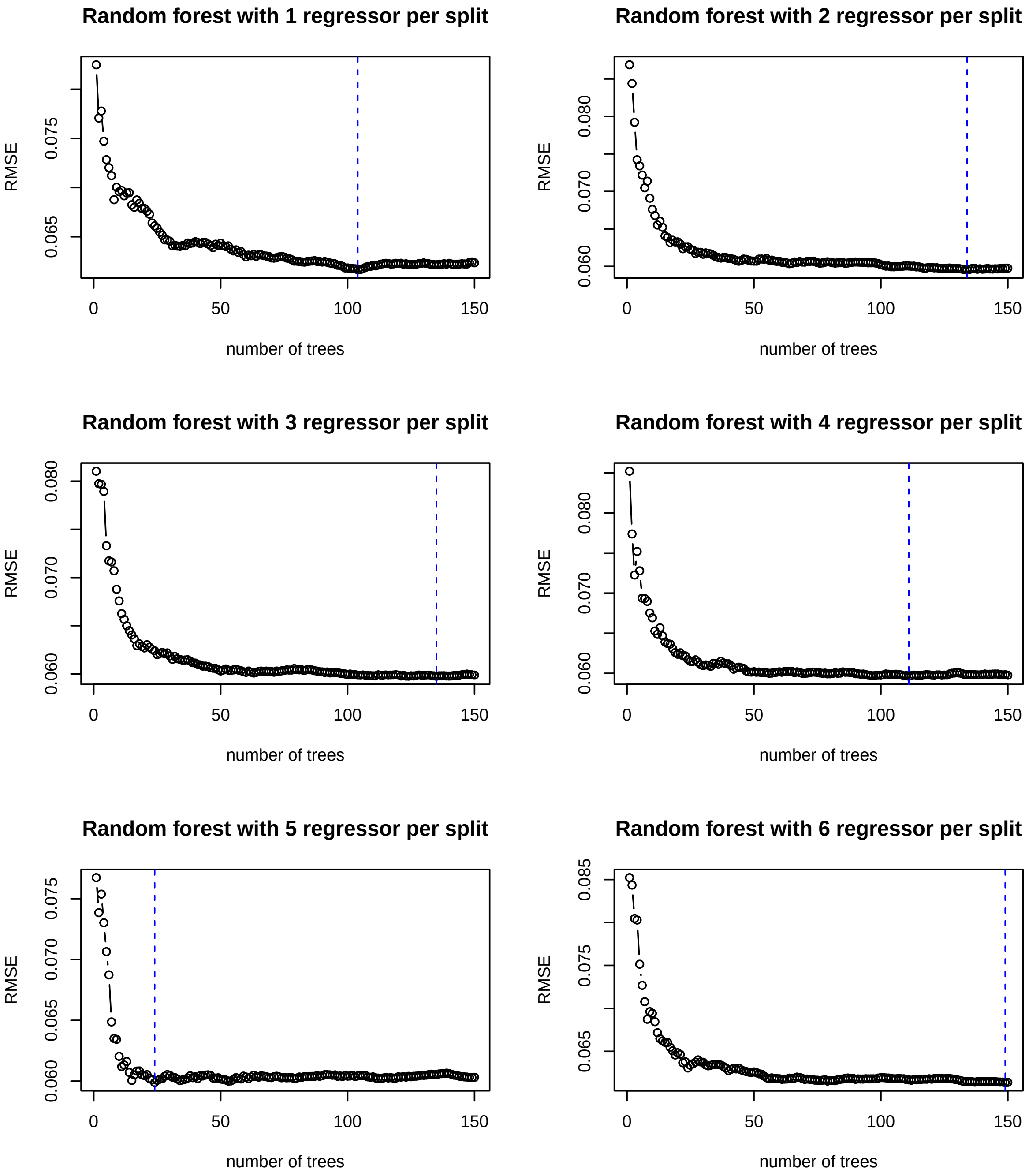


Figure 12: Random forest algorithm with different subset of regressors per split. The blue line indicates the minimum RMSE and the number of tree associated.

After several plots we noticed that the RMSE stabilizes after 100/150 trees, so we set the max trees at 150 for all the models in order to avoid overfitting.

Table 10: Number of regressors used per each split in the random forest approach followed by the RMSE, both for training and validation, and the number of tree associated

nRegressors per slipt	RMSE train	RMSE test	min nTrees
1	0.0616	0.0710	104
2	0.0596	0.0668	134
3	0.0598	0.0661	135
4	0.0596	0.0685	111
5	0.0599	0.0695	24
6	0.0614	0.0709	149

As shown in the table 10 the best model after the optimization is the one that performs 3 splits per tree. We retrieved all the information associated to the model and analyzed the residuals, but they act the same as the begging model.

5.3 Boosting

Boosting model is more complex compared to bagging and random forest. The trees are built using the previous trees' information in order to improve the predictions. The learning rate of the boosting models is controlled by a parameter called, shrinkage parameter λ, usually this parameter tends to be a small value and for this reason the model requires a very large number of trees.

For this analysis we are going to fix a really low learning rate ($\lambda = 0.001$), in this way we can notice if we are overfitting or not. Furthermore, we are also going to try different numbers of depths which are the number of splits made for each tree to see how the models are going to behave. After several tests, the maximum number of trees which overfitting starts is 6000 trees, so we are going to use this as boundary for our models.

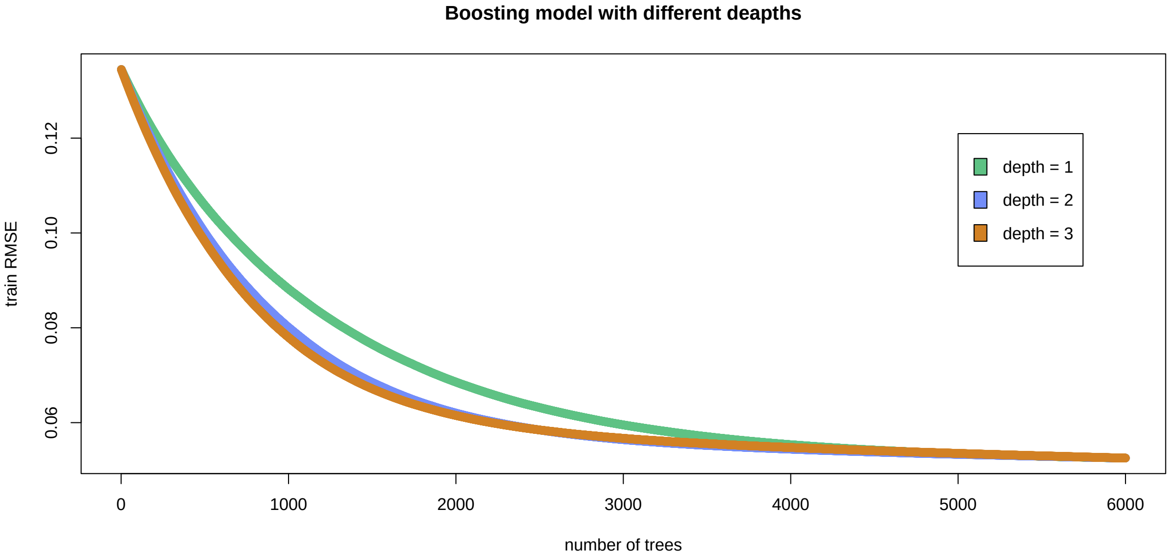


Figure 13: Training 3 boosting models with different depths but fixed number of trees

The models tend to have a similar behavior but with different slops. To select the best model, we are going to compare the "RMSE train" of the models and then see how the final model performs in validation.

Table 11: Boosting models with different depths using 6000 trees to train the models

Depth	RMSE test
1	0.06848
2	0.06859
3	0.06872

The best model in validation is the one with depth = 1 which has the lowest "RMSE test".

5.4 Final considerations about trees

Comparing all the tree approaches, we have the following results:

Table 12: All the best models analyzed and their most important statistics after the optimization

Model	RMSE test	Shapiro-Wilk	nTrees	Split regressors
`prune_model`	0.082	pv = 8.130e-06	1	7
`bagg_model`	0.072	pv = 1.642e-07	90	7
`forest_model`	0.066	pv = 1.835e-08	135	3
`boost_model`	0.068	pv = 7.738e-09	6000	7

The best performing model in validation is the `forest_model` instead the worst one is `tree_model`. The normality distribution of the `tree_model` is the best one compared to the most advance models. In this case the `boost_model` which is the most complex model and the most time expensive is the worst in terms of residual behaviour.

Table 13: Percentage of importance of each regressor. A high values means that the regressor is most frequently used for the splitting phase. Instead the last column of the table is the percentage of variability of the model itself in explaining the dataset, or how good the model is in catching the infomarion inside the dataset.

Model	GRE	TOEFL	UniRatings	SOP	LOR	CGPA	Research	%var
`prune_model`	33.3	0.0	0.0	16.7	0.0	50.0	0.0	
`bagg_model`	10.2	5.1	3.0	3.9	1.3	36.6	2.2	78.7
`forest_model`	11.9	6.1	6.4	5.7	3.3	21.5	4.0	80.1

The regressors variability in the `prune_model` can be seen in Fig. 8, instead regressors variability in the `boosting_model` can't be seen because of the model complexity, but the algorithm itself used all 7 regressors for splitting. Comparing `bag_model` and `forest_model` the variability explained by the models seems to be a little different:

Table 14: Ordering the most important regressors per model starting from the most important.

Model	Importance or regressor
`prune_model`	CGPA > GRE > SOP
`bagg_model`	CGPA > GRE > TOEFL > SOP > UniRatings > Research > LOR
`forest_model`	CGPA > GRE > UniRatings > TOEFL > SOP > Research > LOR

The `bagg_model` considers the regressor TOEFL more important than UniRatings, instead regarding the `forest_model` is vice versa. Besides this little change, the ordering of the other regressors is the same for both models, so we can say that there is consistency in weighing the importance of the regressors in the models.

For our goal in the ensemble methods we lose interpretability especially in the boosting method, whereby the only approach we can consider is the `prune_model`, but we have to take into account that is a non-robust method, this means every time we compute the algorithm we will have a different solution.

6 Conclusion analysis

After all the analysis, the models that reflect our goals are `lm_model` and `prune_model`. The residuals of both models act in the same way following same pattern: negative skewness and heteroscedasticity. The variability left into the residuals that the models don't capture is probably because there is/are significant missing variable/variables inside the dataset, so we can't do much about it.
In validation the `lm_model` performs better with a higher RMSE (which is our comparison parameter) than the `prune_model`. Since the `prune_model` is a non-robust algorithm and performs worse than the `lm_model` we can clearly state that the `lm_model` keeps being the model which better explains our dataset.

For those who are interested the code of this analysis is available on Github [3].

References

[1] James Gareth, Witten Daniela, Hastie Trevor, and Tibshirani Robert. *An introduction to statistical learning: with applications in R*. Spinger, 2013.

[2] Mohan S. Acharya, Asfia Armaan, and Aneeta S. Antony. A comparison of regression models for prediction of graduate admissions, 2019. URL `https://www.kaggle.com/datasets/mohansacharya/graduate-admissions`.

[3] Silviu Filote and Jonathan Bommarito. Statistical learning project, 2023. URL `https://github.com/silviufilote/Statistical-learning`.