Glioma Grading - the Quest for Fewer Features

Emelie Wahlstedt

2024-06

Overview

The dataset that is used for this project is called “Glioma Grading Clinical and Mutation Features” and was accessed via the UC Irvine Machine Learning Repository. The aim of this project is to find the fewest features possible to accurately predict whether a glioma is a Lower-Grade Glioma (LGG) or a Glioblastoma Multiforme (GBM).

The fewest possible number of features is desired as molecular testing is expensive - most of the features in the dataset are “molecular features” that are either mutated, or not. Each molecular feature will be costly to test for mutations. We want to achieve a high accuracy, but still try to save on testing resources.

A research paper by Tasci et. al using this dataset had a similar aim; minimal features and maximum accuracy. The research team used both feature selection methodologies (one novel methodology developed by the team) and ensemble modelling. They achieved an accuracy of 87.606% - we will aim for this and see how we do with our own models.

Two files are available from the repository: the original dataset (862 observations), and a preprocessed file (839 observations). We will use the preprocessed file - the preprocessing steps are described as follows by the donors of the dataset:

> *“The original and preprocessed files differ in the following ways:*
>
> - *There are 23 instances in the original file where Gender, Age_at_diagnosis, or Race feature values are ‘–’, or ‘not reported’. These instances were filtered out in the preprocessed dataset.*

- *Despite being present in the original dataset, we do not include the columns Project, Case_ID, and Primary_Diagnosis columns in the preprocessed dataset.*
- *Age_at_diagnosis feature values were converted from string to continuous value by adding day information to the corresponding year information in the dataset as a floating-point number for the preprocessing stage."*

"Below is a list of the additional columns of the original dataset file (and their corresponding description):

- *Project column represents corresponding TCGA-LGG or TCGA-GBM project names.*
- *Case_ID column refers to the related project Case_ID information.*
- *Primary_Diagnosis column provides information related to the type of primary diagnosis."*

To keep things clear, this table shows what has happened to each column in the original file during preprocessing:

Table 1: Changes made to variables of original dataset in minimal dataset

Variable	Key
Grade	0 = LGG 1 = GBM
Project	Removed
Case_ID	Removed
Gender	0 = Male 1 = Female
Age_at_diagnosis	Converted from string to continuous value
Primary_Diagnosis	Removed
Race	0 = white 1 = black or african American 2 = asian 3 = american indian or alaska native
All columns representing molecular features	0 = NOT_MUTATED 1 = MUTATED

Data exploration

In order to better understand the dataset and also aid our decision making for feature and model selection, we do a deep dive into the dataset. We start by looking at the distribution of the different feature variables across the dataset, and then look at their relationship with our target variable, Grade.

Distibution of data

Table 2: Distribution of gender in the minimal dataset

Gender	Count	Percent
Female	351	41.84
Male	488	58.16

The distribution between female and male appears to be relatively even, with male being slightly overrepresented.

Table 3: Distribution of race in the minimal dataset

Race	Count	Percent
American Indian or Alaska Native	1	0.12
Asian	14	1.67
Black or African American	59	7.03
White	765	91.18

Over 90 percent of the data points belong to the race “white” - this means that race as a feature in this dataset will need to be used with caution, as the numbers for other races are low. It will also be difficult to divide the dataset into train and test sets while keeping the datasets even and representative, since there is only one American Indian or Alaska Native in the dataset and not many cases in the other categories either.

Table 4: Distribution of grade in the minimal dataset

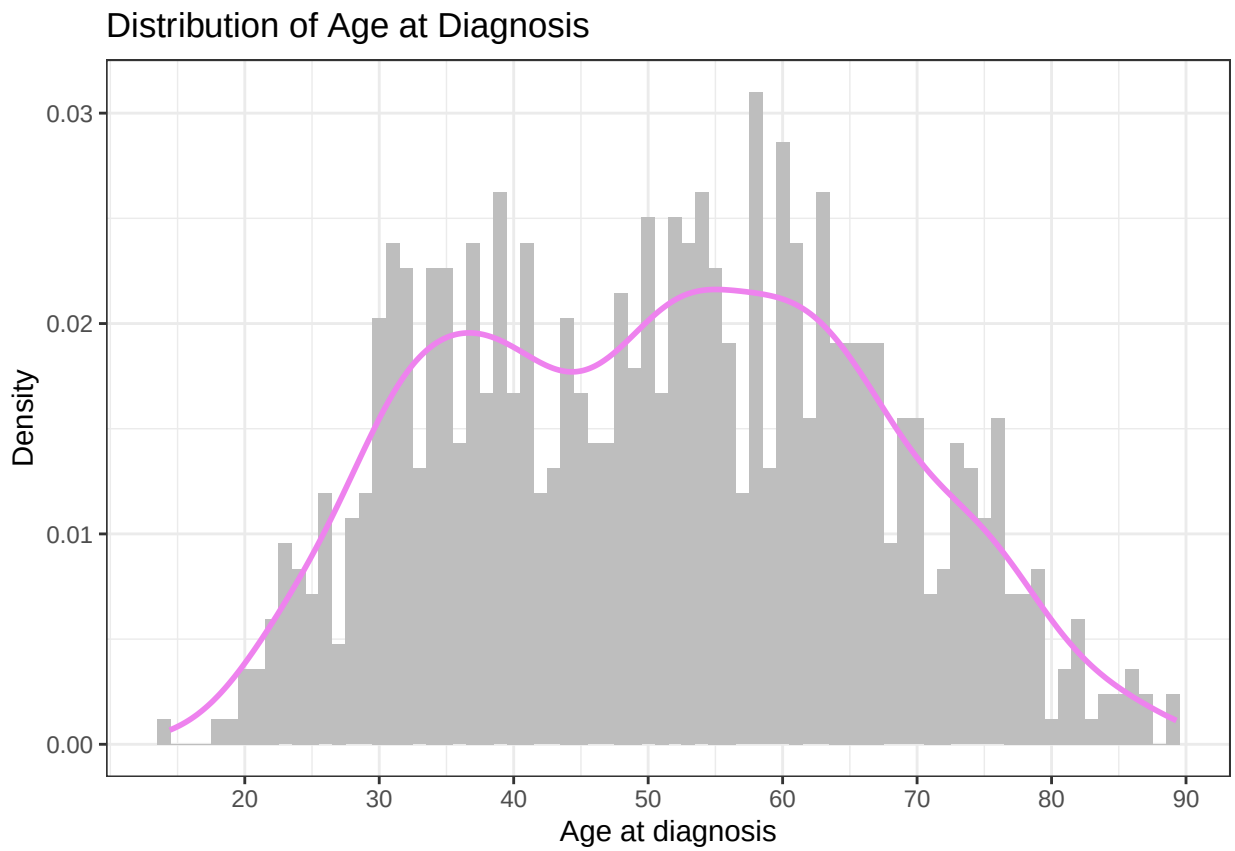
Grade	Count	Percent
GBM	352	41.95
LGG	487	58.05

The grade variable, like gender, seems to be relatively evenly distributed across the dataset, with LGG slightly more common.

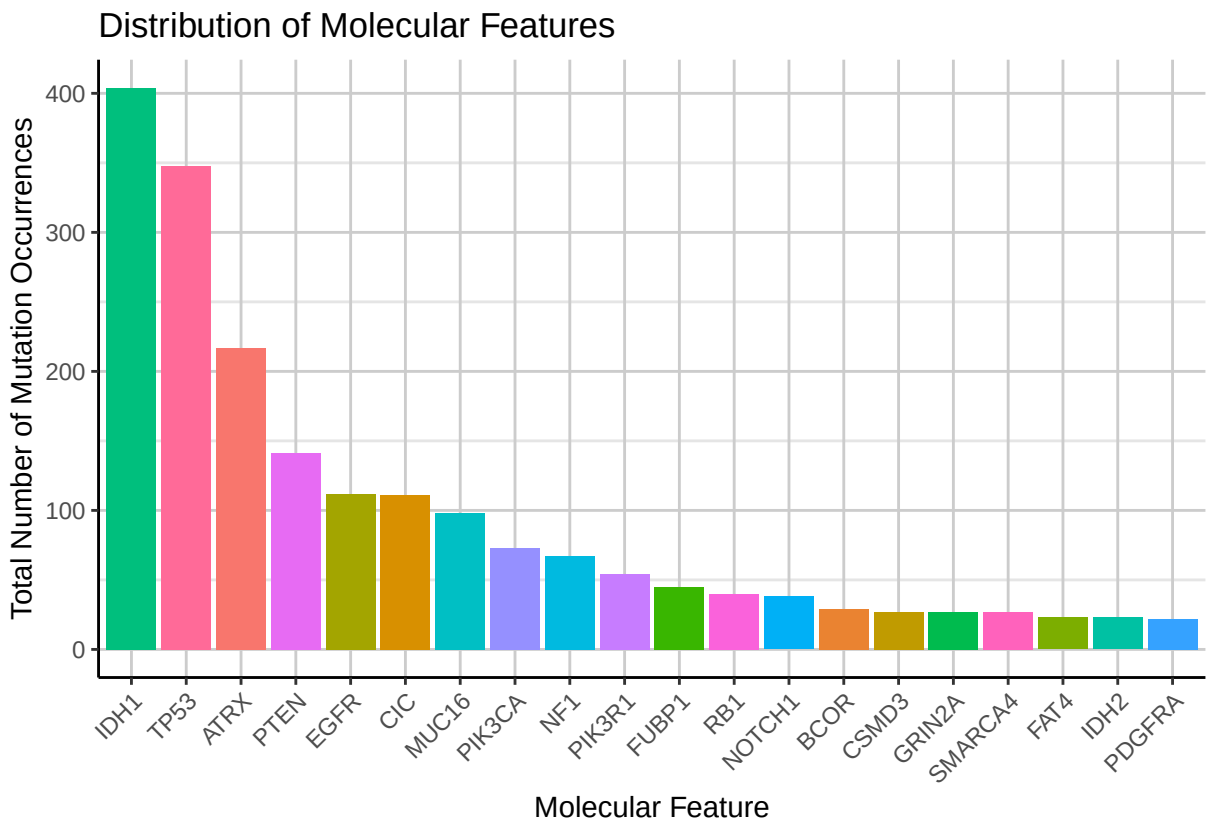
Table 5: Distribution of age at diagnosis in the minimal dataset

Statistic	Value
Mean	51
Median	52
Youngest	14
Oldest	89

There is a fairly wide spread of ages in the dataset, which the table does not quite tell enough about. Plotting the spread of ages may be more informative:



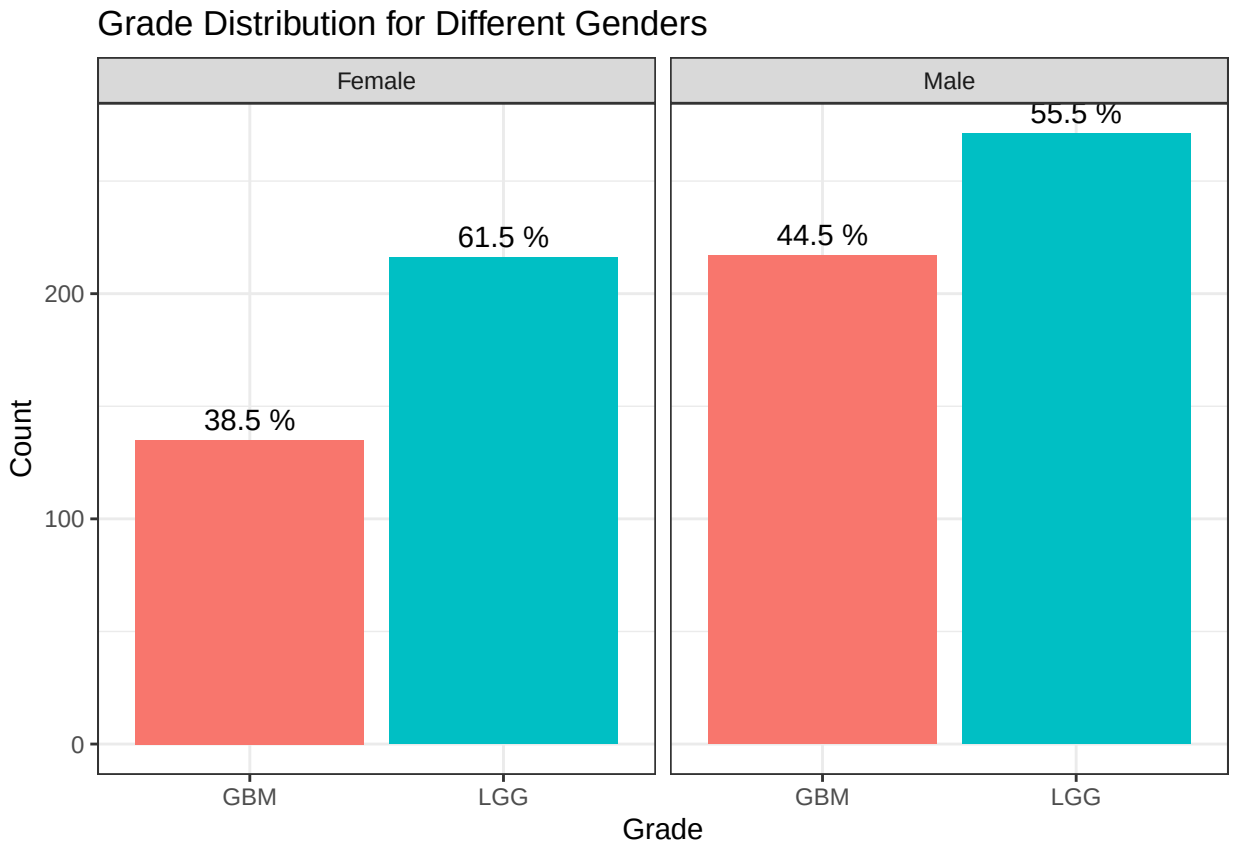
There appears to be a dip in the age distribution around age 45, somewhat dividing the age groups into two. This could mean that age at diagnosis may be useful for prediction, but we need more information before we can draw this conclusion.



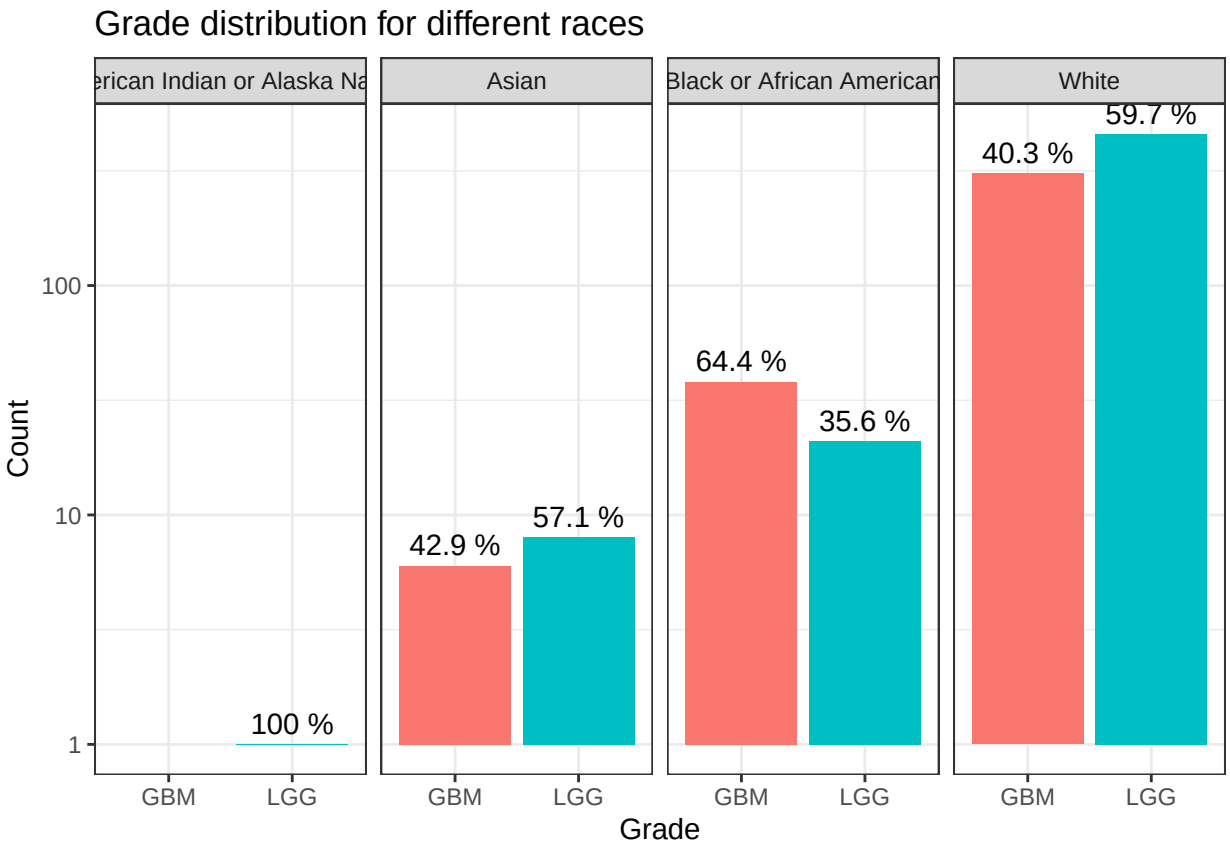
We can see that some mutations are much more common than others. In order for our models to be relevant, it would seem logical to focus on features that are likely to be present in the majority of glioma cases.

The relationships of the Grade variable and the features

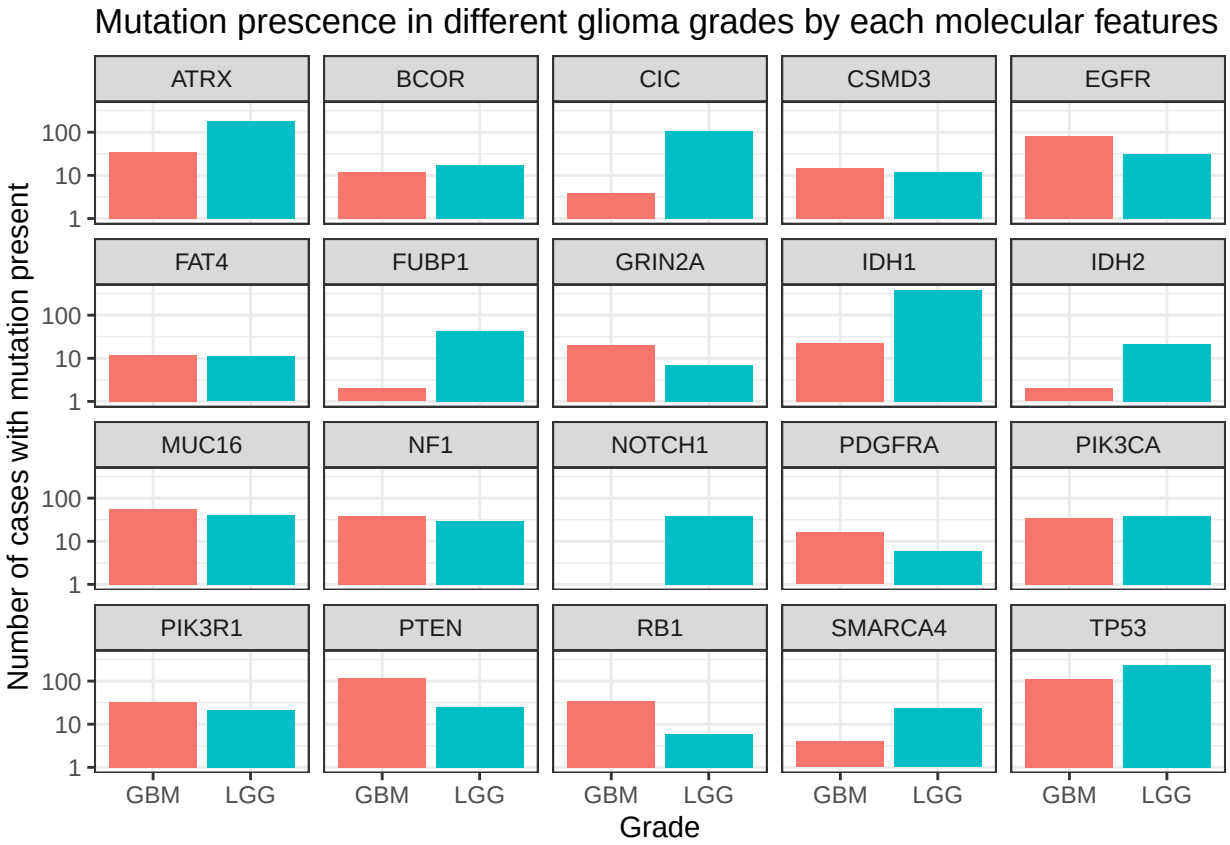
It will also be useful to study the relationships between the Grade variable, which will be the target variable in our model, and the feature variables.



Starting with the Gender variable, it appears that there is a slightly bigger difference in Grade for females than males, but the difference is fairly small. Nevertheless, Gender may be a useful in predicting Grade to some extent.



When looking at the Race variable, it is important to remember that the majority of the dataset belongs to the same race (white). The apparent difference in proportion of Grade for black or African american compared to white and Asian may be a true difference, but since the sample size here is very small (59 total cases compared to 795 for white) it needs to be interpreted with caution.



Some molecular features show larger differences than others between mutated or not mutated, but this is a log scale - let's see where the biggest real differences are:

Table 6: Differences in presence of mutation of molecular features in cases of LGG and GBM

mol_feats	GBM	LGG	Diff	Pct_diff
NOTCH1	0	38	38	100.00
CIC	4	107	103	92.79
FUBP1	2	43	41	91.11
IDH1	23	381	358	88.61
IDH2	2	21	19	82.61
SMARCA4	4	23	19	70.37
RB1	34	6	28	70.00
ATRX	34	183	149	68.66
PTEN	116	25	91	64.54
GRIN2A	20	7	13	48.15
PDGFRA	16	6	10	45.45
EGFR	81	31	50	44.64
TP53	113	235	122	35.06
PIK3R1	33	21	12	22.22
BCOR	12	17	5	17.24
MUC16	57	41	16	16.33
NF1	38	29	9	13.43
CSMD3	15	12	3	11.11
PIK3CA	34	39	5	6.85
FAT4	12	11	1	4.35

The molecular features for which there is a large difference in presence of mutation between the two grades may be more important features for building a predictive model. In this case the biggest difference between grades appears to be the molecular feature NOTCH1 - this does not seem to be a very common mutation, however.

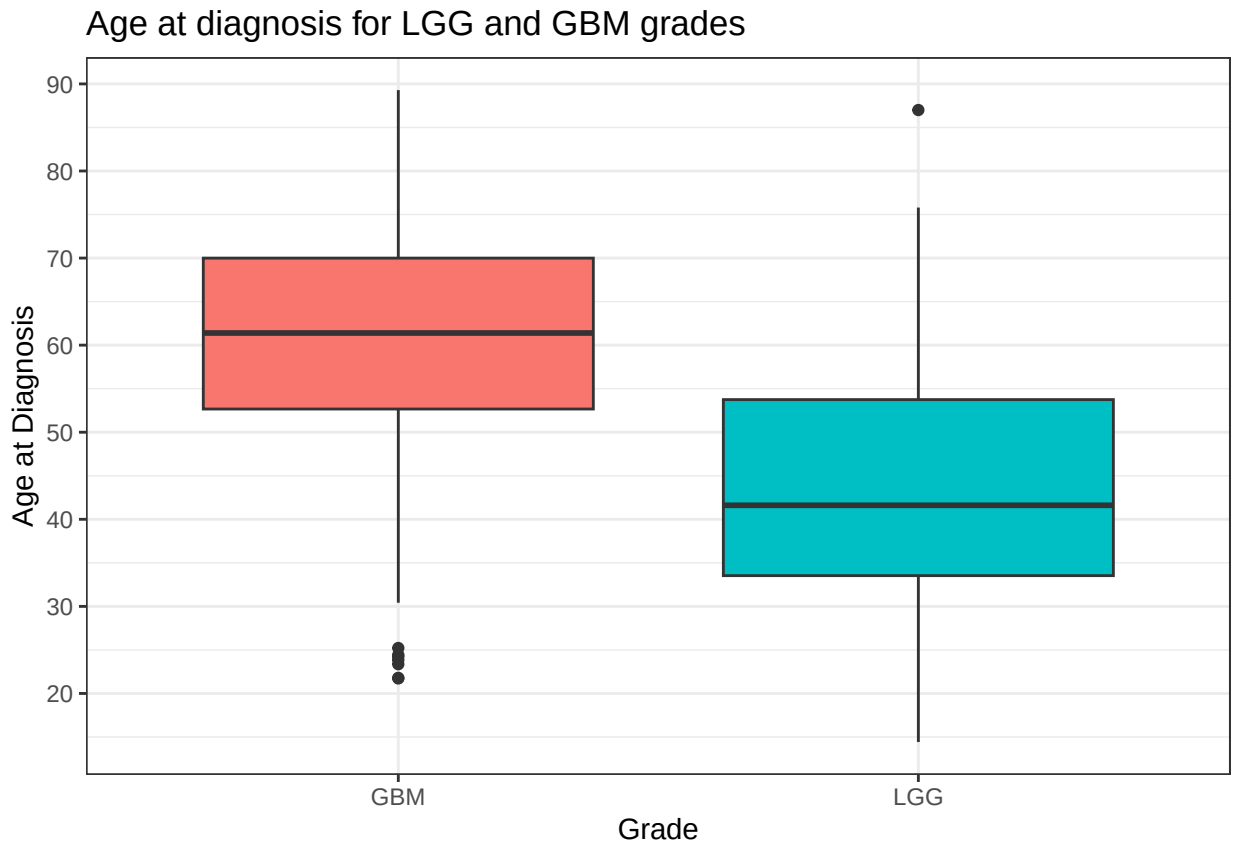
We can look at the table again with only mutations that occur at least 100 times in the dataset:

Table 7: Differences in presence of mutation of most common molecular features in cases of LGG and GBM

mol_feats	GBM	LGG	Diff	Pct_diff
CIC	4	107	103	92.79
IDH1	23	381	358	88.61
ATRX	34	183	149	68.66
PTEN	116	25	91	64.54
EGFR	81	31	50	44.64
TP53	113	235	122	35.06

Features that may be very useful to us, since they appear to be relatively common and also have a big difference in mutation between GBM and LGG, are CIC, IDH1, ATRX and PTEN. IDH1 is the most commonly mutated feature and therefore it will probably be the most useful. PTEN is the only feature of the ones above that is more commonly mutated in GMB than LGG, which could also be useful.

Let's also look at age at diagnosis:



There is only a small overlap of the interquartile ranges, so there may be something in age at diagnosis being a useful predictor of grade.

We have looked at all of the different features now - some seem to hold more information than others. We will further look into which features will be most useful to us when we build our models.

Method

Create train and test sets from the minimal_dataset

To gain some more insight into data partition in the context of smaller datasets and medical data, the internet was consulted. In their blog, Draelos describes the need to ensure that the data is split by patient, not by encounter - as if the same patient appears twice, their data will be highly correlated. We can check the original file for any duplicates by checking the CaseId:

Table 8: Comparing length of unique CaseId and all CaseId in unprocessed dataset

Length_unique	Length_all
862	862

The unique values and the full vector length are the same, so there should not be any patients included more than once.

The suggestion from the dataset donors is also to use cross-validation for small datasets rather than a train-validation-test setup, as creating both validation and test sets will make each set too small to be useful. We will work cross-validation into our models where possible - the dataset creators suggest 10-fold cross-validation.

There are many suggestions out there on test-train splits, the most common seemingly being 80-20 and 70-30. Since the dataset is relatively small, we will go for 70-30 to try to obtain sets that are broadly representative of the whole dataset.

```
set.seed(1) # Set seed for reproducibility
minimal_dataset$Grade <- ifelse(minimal_dataset$Grade == 0, "LGG", "GBM") # Change back
 ↪  to abbreviations for interpretability
minimal_dataset$Grade <- factor(minimal_dataset$Grade) # Turn into factor for model
 ↪  building

test_index <- createDataPartition(y = minimal_dataset$Grade, times = 1, p = 0.3, list =
 ↪  FALSE) # Create index for 30 percent of data
```

```
train_set <- minimal_dataset[-test_index, ] # Select rows not in test index (70 percent
↪  of data)
test_set <- minimal_dataset[test_index, ] # Select rows in test index (30 percent of
↪  data)
```

Table 9: Distribution of the Grade variable across training and test sets compared to full dataset

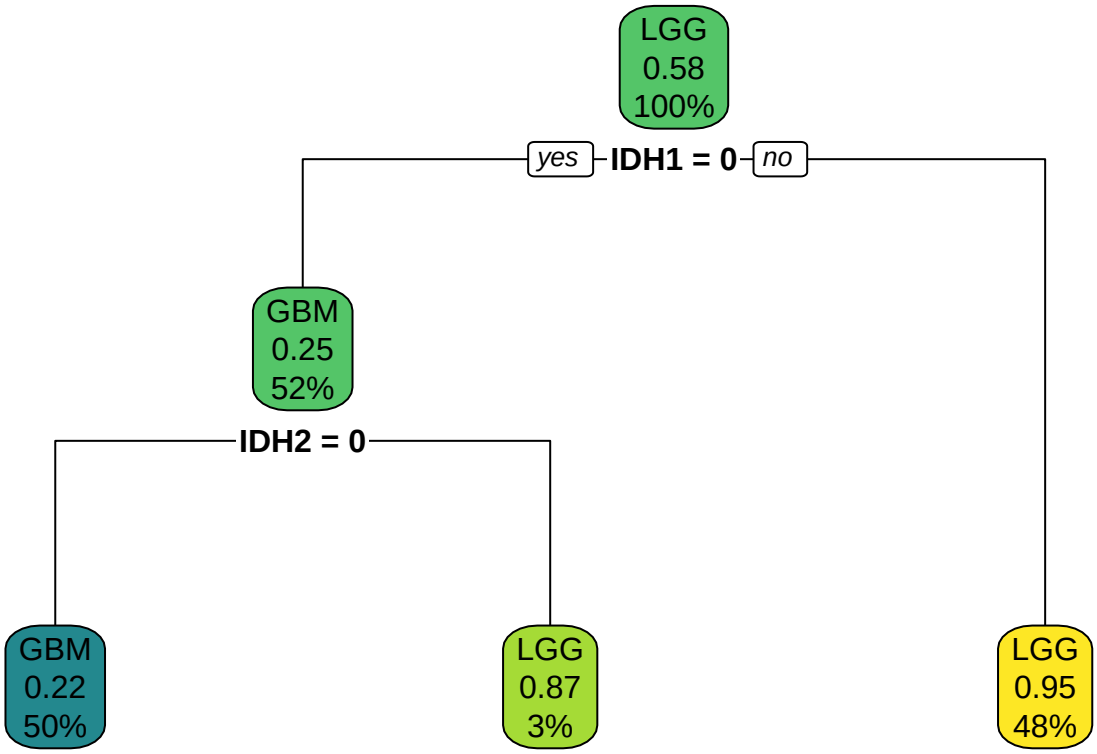
Set	Grade	Count	Percent
Full	GBM	352	42.0
	LGG	487	58.0
Train	GBM	246	42.0
	LGG	340	58.0
Test	GBM	106	41.9
	LGG	147	58.1

It seems like the sets have been divided roughly equally with regards to Grade. The function createDataPartition which was used is built to create balanced splits, so the check above reassures us that it works reasonably here.

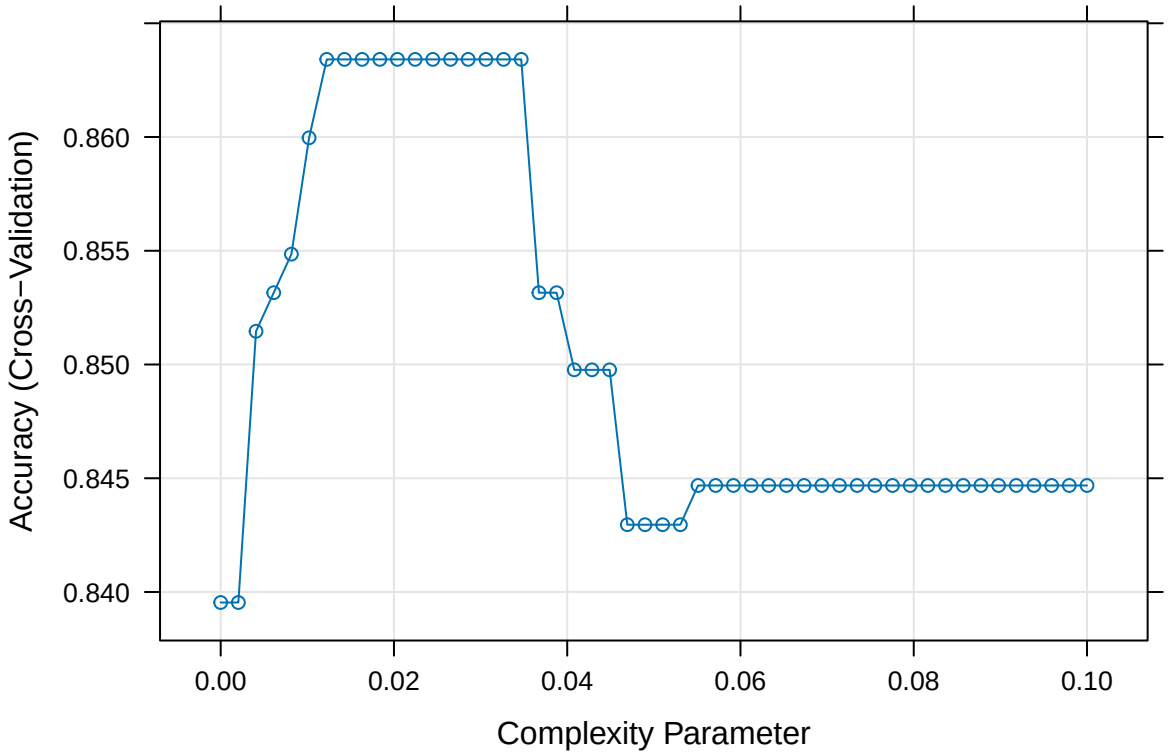
Model testing

Decision Tree model with rpart

As most of our variables are binary, a decision tree approach may fit the data well. We have seen that some molecular features are mutated more or less frequently for the two different glioma grades, so there may be some easy decisions that can be made using some of the molecular features. We test the rpart model to build a decision tree using all features as a starting point:



This is interesting - like we suspected when exploring the data, IDH1 is an important feature, and a relatively common mutation from what we can tell from the dataset. IDH2, which is not very commonly mutated (23 in total) came up as significant for the second split in the tree - while potentially useful, this needs to be treated with caution, as it is perhaps unlikely to be present in a large number of cases in the real world.

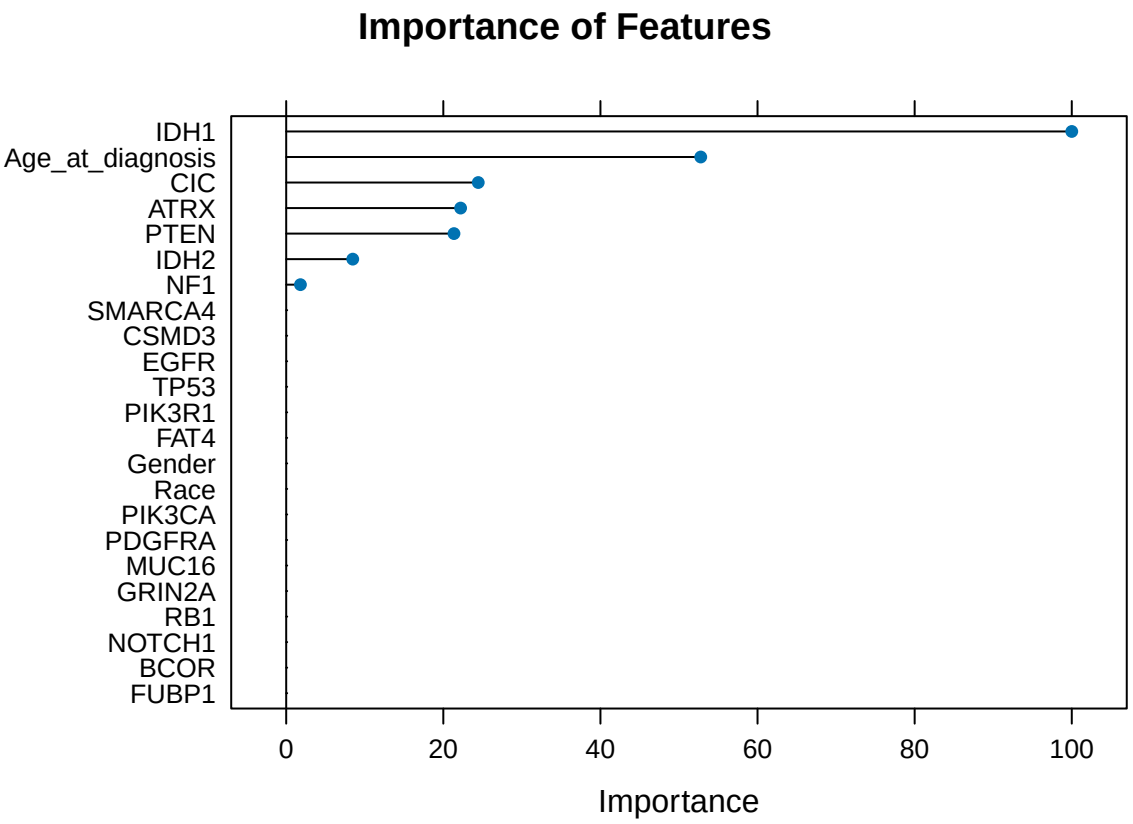


Accuracy and confusion matrix for the rpart model:

0.8814229

Prediction	Reference	
	GBM	LGG
GBM	98	22
LGG	8	125

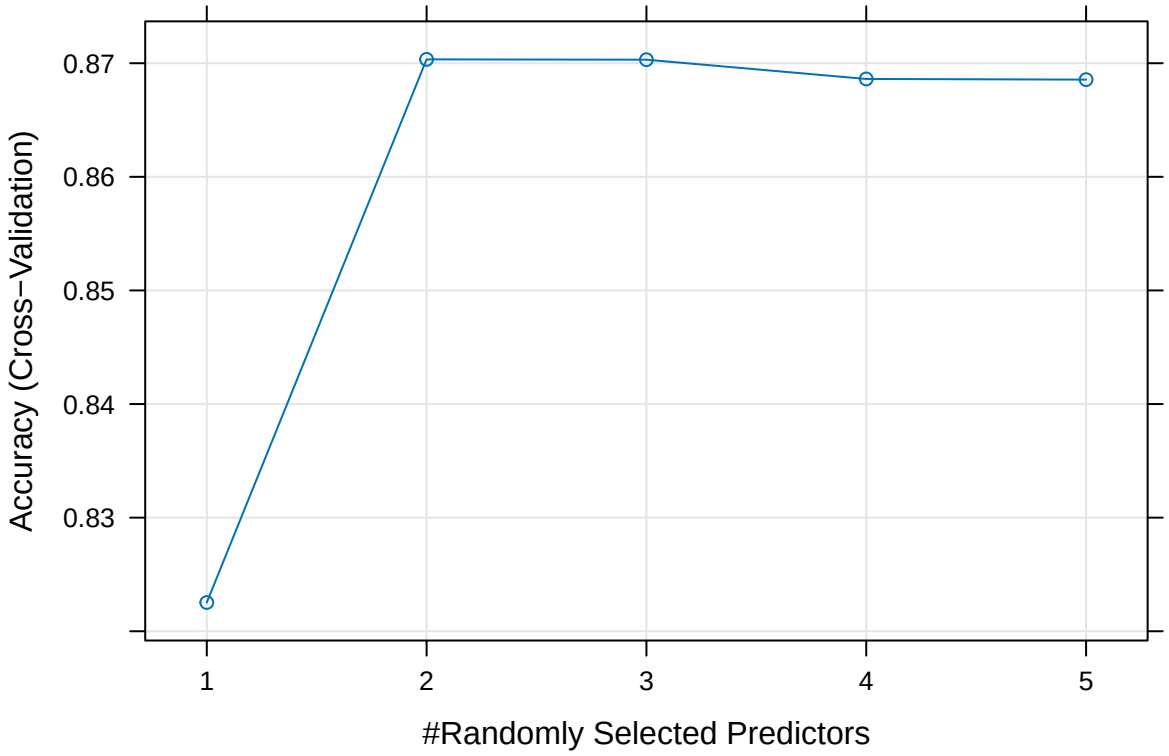
The accuracy we get with this model is actually not bad, 0.8814229. We can see from the confusion matrix below that the sensitivity is quite high, while the specificity is slightly lower. Compared to the ensemble models that Tasci et. al created, this is actually pretty good - but we are still using all of our 23 features, and we want to achieve this level of accuracy with fewer than this.



Looking at the most important features, we can see that as expected IDH1 comes out on top. For this model, only seven out of the total 23 features appears to have any importance at all.

Random forest model

While decision trees are lovely and simple to interpret, they can be prone to overfitting. There are however machine learning models that can produce more robust results, such as the random forest model. This is also a tree-based model, but it combines many different decision trees to obtain a result, essentially a form of ensemble modelling.



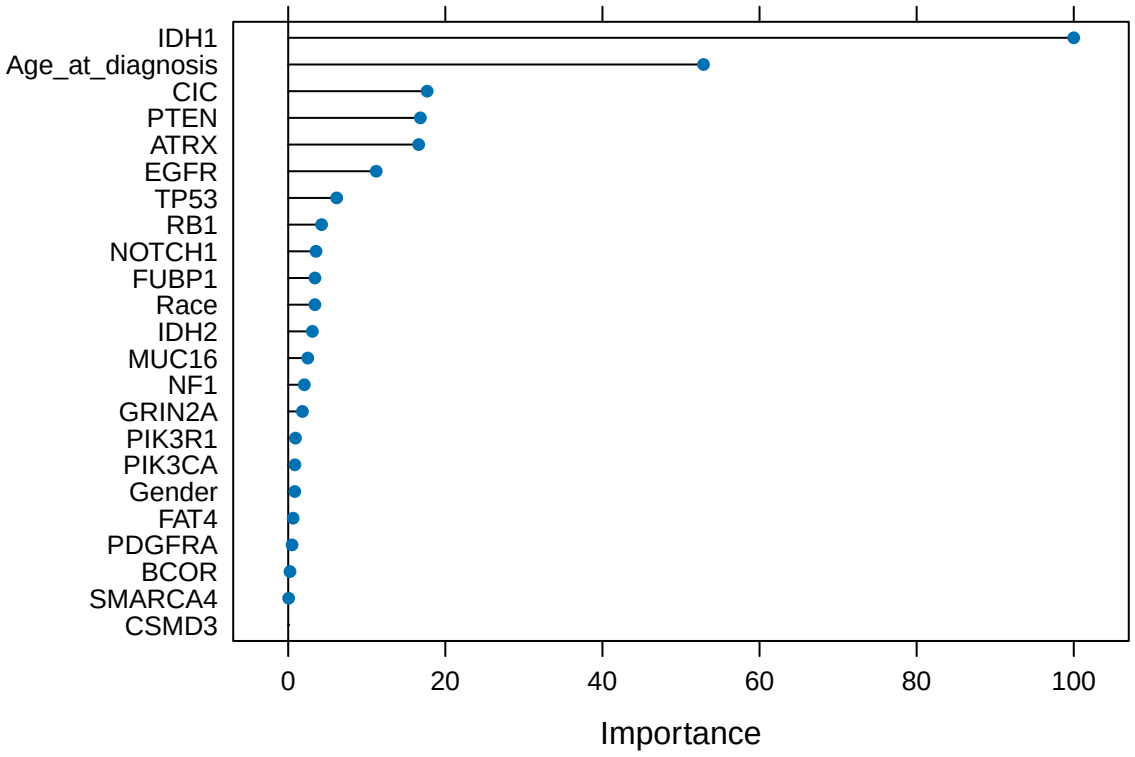
Accuracy and confusion matrix for the random forest model:

0.8853755

Prediction	Reference	
	GBM	LGG
GBM	96	19
LGG	10	128

We can see that the more features are chosen, the less accurate the model becomes - this makes sense as we saw in the tree model earlier how most features have little to no predictive value.

The random forest model also allows for the feature importance to be examined:



This model certainly implies that some features are less important, but unlike the rpart tree model it does seem to assign some importance to all of the features, albeit small. IDH2 is given less importance in this model, similarly to when we looked at the differences between the molecular features during data exploration.

XGboost model

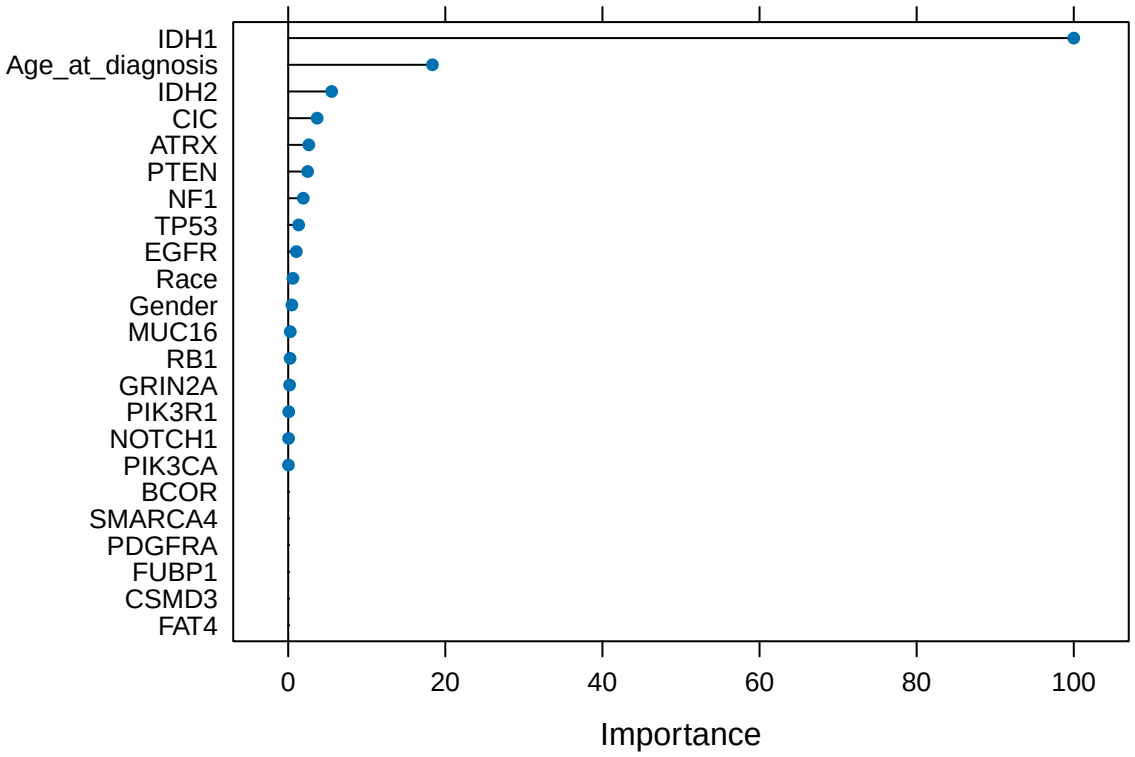
It may be useful to try another model - the xgboost model is an “optimized distributed gradient boosting library” . Simply speaking, “gradient boosting” is a machine learning technique which learns from previous mistakes - for every round that the algorithm runs, it builds a new model which is an improvement from previous ones. The final output is an ensemble of all of the models, weighted by how much each model can reduce prediction error.

Accuracy and confusion matrix for the xgboost model:

0.8853755

Prediction	Reference	
	GBM	LGG
GBM	97	20
LGG	9	127

The accuracy we get with this model is quite good, but similar to the other models.



Plotting the importance of the features again confirms that IDH1 is the most important feature, followed by age at diagnosis.

Feature selection

After running three different models, it is clear that IDH1 is the most important feature. However, relying on just one feature will probably not give us a very high accuracy. After all, Tasci et al. actually used 15 (14.9) features to get their results.

Another thing we can do to help us decide on features for our model is to check for features with near-zero variance, in other words where there is very little or no information provided by a feature. The caret package has a function for this called nearZeroVar - we run this function on the minimal dataset and extract the names of the features that it identifies:

Table 10: Features with near-zero variance

RB1
NOTCH1
BCOR
CSMD3
SMARCA4
GRIN2A
IDH2
FAT4
PDGFRA

These variables have very little variation and is therefore deemed by the function to be unlikely to be significant for us - however, IDH2 is on this list, which was significant for the decision rpart tree model. We clearly need to be careful in dismissing features as not useful based on their variance.

We can do a very simplified voting system, not as advanced as the one that Tasci et al. created for feature selection but it will have to do. Let's take the top five features of each model and see where we end up:

Table 11: rpart model top five features

	Overall
IDH1	100.00000
Age_at_diagnosis	52.77251
CIC	24.44636
ATRX	22.20479
PTEN	21.35818

Table 12: random forest model top five features

	Overall
IDH1	100.00000
Age_at_diagnosis	52.86729
CIC	17.68829
PTEN	16.82717
ATRX	16.61412

Table 13: xgboost model top five features

	Overall
IDH1	100.000000
Age_at_diagnosis	18.357974
IDH2	5.531400
CIC	3.686319
ATRX	2.625060

Let's include all of the features in the tables above - this gives us:

- IDH1
- Age_at_diagnosis
- CIC
- ATRX
- PTEN
- IDH2
- NF1

Re-running models with fewer features

Let's see how our models work with the selected features.

Table 14: Accuracy of all models tested so far

Model	Accuracy
random forest - all features	0.8853755
xgboost - all features	0.8853755
rpart - all features	0.8814229
rpart - top features	0.8814229
random forest - top features	0.8774704
xgboost - top features	0.8774704

In summary, we can see that the models perform generally worse with the selected features than with all of them. Worth noting here is that several of the accuracies have come out exactly the same - this is likely due to the dataset being very small. This also puts into question how effective our cross-validation is, and whether the models will be scalable to larger datasets.

Ensemble model

Tasci et al. used a number of different ensemble models in their work. Ensemble models can be more robust than single models, as they are able to take into account the strengths and weaknesses of different models to supposedly give a better prediction. We create a relatively simple version, a majority vote ensemble, and set it up as follows:

```
# Majority voting ensemble
ensemble_vote <- cbind(rpart = pred_rpart_top == "LGG",  rf = pred_rf_top == "LGG",
↪  xgboost = pred_xgboost_top == "LGG")

# Predict on test set
pred_ensemble_vote <- ifelse(rowMeans(ensemble_vote) >= 0.5, "LGG", "GBM")
```

```r
# Compute and save accuracy
acc_ensemble_vote <- mean(pred_ensemble_vote == test_set$Grade)
```

Results

Table 15: Accuracy of all models tested including ensemble model

Model	Accuracy
Ensemble - voting with top features	0.8932806
random forest - all features	0.8853755
xgboost - all features	0.8853755
rpart - all features	0.8814229
rpart - top features	0.8814229
random forest - top features	0.8774704
xgboost - top features	0.8774704

We can see that in this case ensembling of the models works pretty well - individually, the models with fewer features are generally outperformed by the ones with more features, but by creating an ensemble we are able to reach a pretty decent accuracy. And we have only used seven features (one of which, age at diagnosis, is “free”; the other six requires molecular testing), which has to be considered pretty good.

Conclusion

As concluded above, the models above work fairly well compared to the models by Tasci et al. that we set out to match. However, there are some issues that point to that the ensemble model that gave the best accuracy for this dataset may not work as well on another dataset:

- The dataset is small, meaning there may be biases within it that are not applicable to a broader population
- The dataset mostly consists of white people - Tasci et al. had trouble applying their methods to a dataset with predominantly Chinese people, suggesting that there may be differences between different populations.
- The accuracies obtained from the different models are similar, and for some models identical. This can occur when the dataset is small and the outcome is binary, which is the case here. What this may mean is that the models are somewhat overfitted, or that there is a very clear trend in the dataset that all the models are picking up in the same way. If this trend is present in a larger population it may be fine, but it could also be a coincidence in this specific population.

It would be interesting to use the ensemble model on a larger and more diverse dataset to see if it holds; I have not managed to obtain the dataset used by Tasci et al. for additional testing, but it would be interesting if this could be achieved in the future.

Final remarks

I have used relatively simple methods to achieve the goal here - there are obvious limitations to how much a relative novice in data science can do. However, the learning along the way has been very valuable and I am looking forward to learning more about methods such as Singular Value Decomposition and other matrix related models, which I have yet to master to a level that made me confident to include them in this report. I also want to learn about other models and approaches that I have come across during this the data science programme in general and during this project in particular, such as Support Vector Machines and also other feature selection methods like recursive elimination.

Acknowledgements

The great people of the internet have certainly helped me in furthering my knowledge about different models, why my code doesn't work, and approaches to solving issues along the way. Forums like stackOverflow and similar websites have been very helpful.

I have used chatGPT for help on a few issues along the way, mostly code error related. I have found it a really useful tool in my learning and also a great moral support when I despair. I have included a full list of issues that I asked for chatGPTs input on in the appendix.

Appendix

chatGPT queries and learning takeaways

- Asked chatGPT for help when I struggled to get table-variables-reference to knit, as I couldn't find answers on any of the forums. chatGPT advised to shorten the very long row text for "Age_at_diagnosis" and also to add "escape = TRUE" to the kbl function.
- Asked chatGPT about the createDataPartition function which was throwing an error, needed to include list = FALSE which I had missed.
- Asked chatGPT why my code for turning Grade variable into its original labels and then into a factor didn't work, helped me fix my code where I was trying to use the pipe when not appropriate to do so.
- Asked chatGPT why I couldn't find the variable importance for my rpart model - I was calling it wrong, varImp is a function, not an object within the model.
- Asked chatGPT how to make my rpart importance table look nicer, since I couldn't figure out how to properly convert it to a data frame. I was doing it wrong by trying to convert the whole object rather than just the importance part of it. Now looks great with kable.
- Asked chatGPT about the xgboost model and how it actually works, and what the different hyperparameters do in the model. This was not a model that was covered in the Data Science programme but I had come across it in my research on various forums, and wanted to test it out. I had a basic model built before asking chatGPT for help but was having trouble with understanding the hyperparameters, and was keen for the model to work with caret::train as I like the format.
- Asked chatGPT to help me arrange my tables across the page in the Feature Selection section - learnt about latex minipages