Student Outcome Prediction

Predicting student's failure to meet passing criteria in online Artificial Intelligence course

Problem Statement

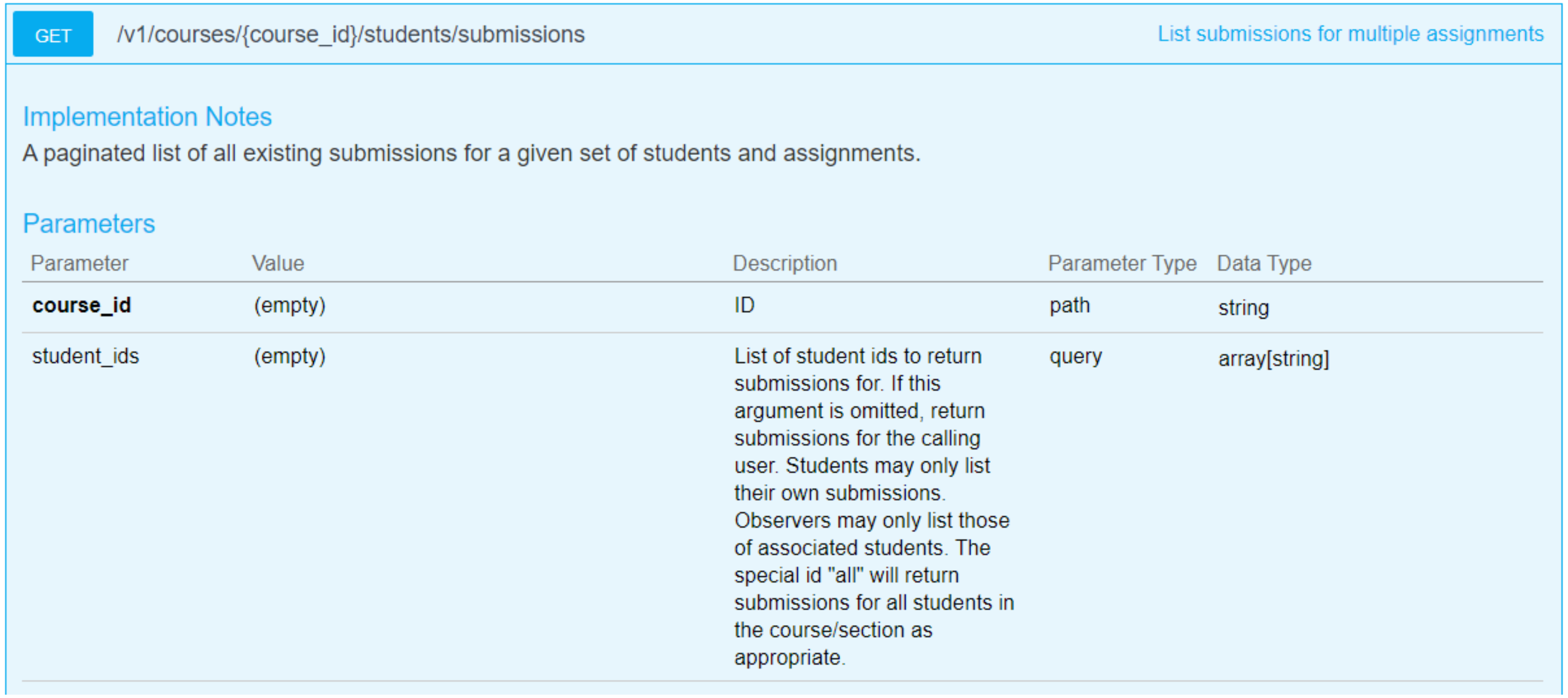
Use historical data from previous students that took the course to predict their likelihood of failing to meet the passing criteria.

To achieve this we will use data from only the first half of the course (4 weeks).

Predicting the probability of a student failing at this stage will allow the course support staff, enough time to reach out to the student and help him get on track to complete.

Data – Outcome Variable

To collect data about the student's outcome we collect all the assignments for each student and their grade. If the student meets the passing criteria for the course, we encode the outcome value to 1, a failure will be encoded as a 0.



GET /v1/courses/{course_id}/students/submissions

List submissions for multiple assignments

Implementation Notes

A paginated list of all existing submissions for a given set of students and assignments.

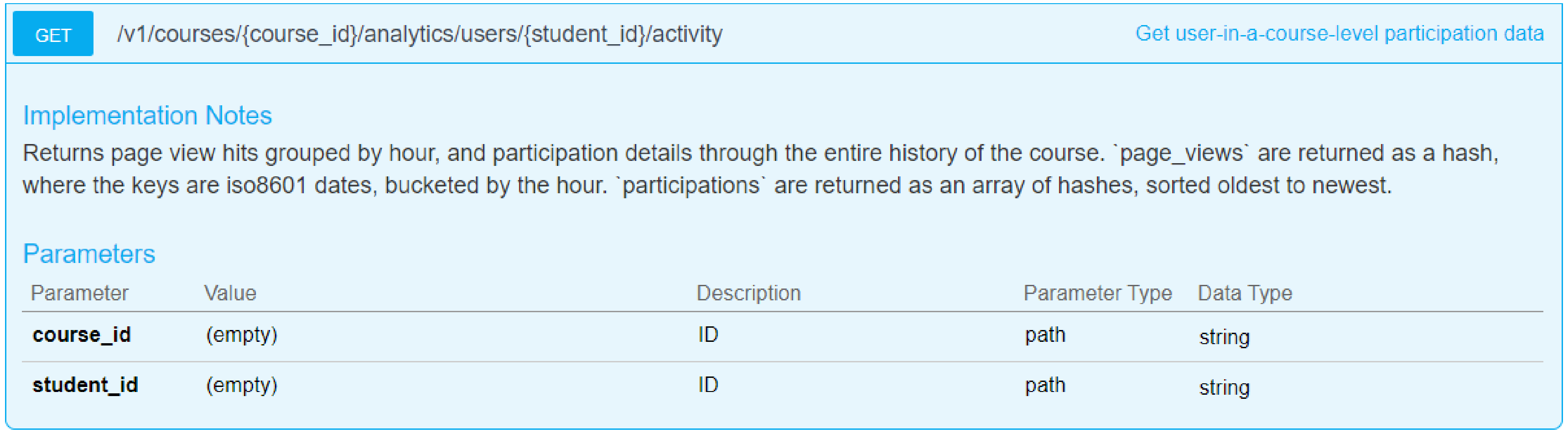
Parameters

Parameter	Value	Description	Parameter Type	Data Type
course_id	(empty)	ID	path	string
student_ids	(empty)	List of student ids to return submissions for. If this argument is omitted, return submissions for the calling user. Students may only list their own submissions. Observers may only list those of associated students. The special id "all" will return submissions for all students in the course/section as appropriate.	query	array[string]

Data – Predictor Variables

There are a number of sources for features. But not many contain a timestamp that we can use to filter data only for the first half of the course

From the course analytics endpoint we can obtain observation about daily page views and participations (Discussion forum entries, assignments submissions, and messages to the faculty).



GET /v1/courses/{course_id}/analytics/users/{student_id}/activity Get user-in-a-course-level participation data

Implementation Notes

Returns page view hits grouped by hour, and participation details through the entire history of the course. `page_views` are returned as a hash, where the keys are iso8601 dates, bucketed by the hour. `participations` are returned as an array of hashes, sorted oldest to newest.

Parameters

Parameter	Value	Description	Parameter Type	Data Type
course_id	(empty)	ID	path	string
student_id	(empty)	ID	path	string

Feature engineering

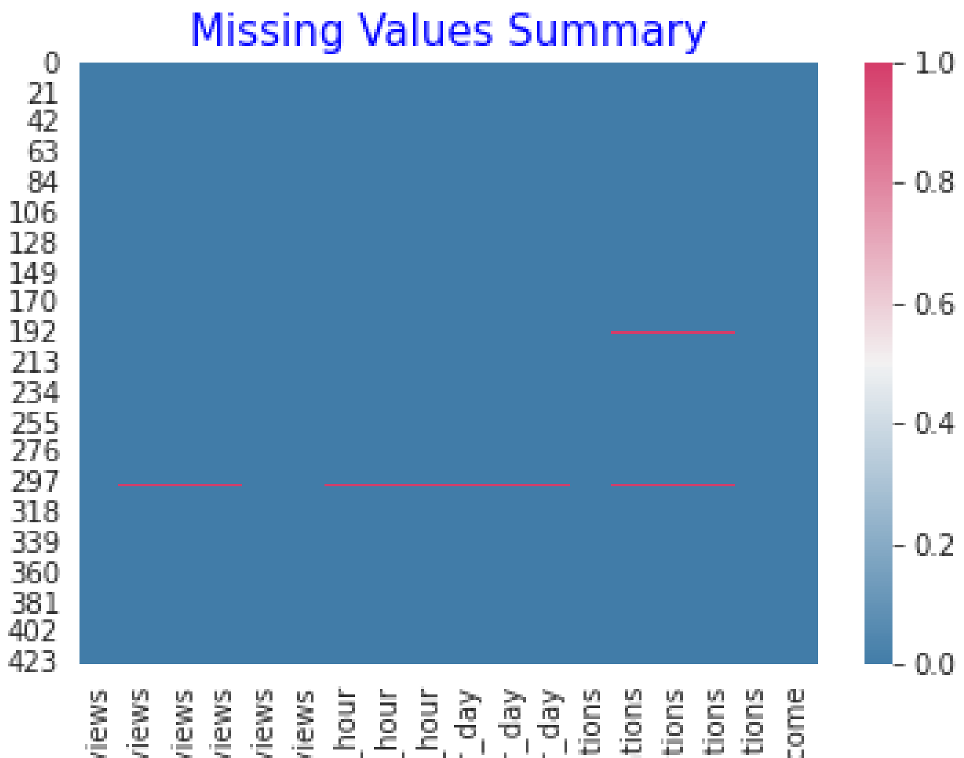
From these two time-series we calculate the following variables with simple permutations and descriptive statistics:

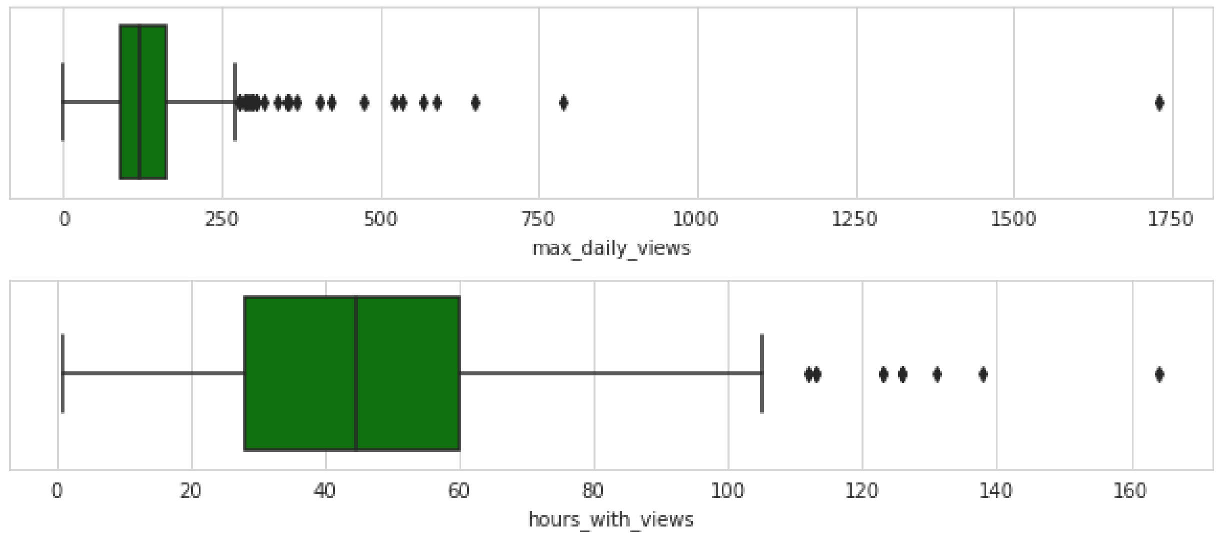
```
'student_id'
'course_id'
'tot_page_views'
'average_daily_views'
'median_daily_views'
'max_daily_views'
'days_with_views'
'hours_with_views'
'max_views_per_hour'
'avg_views_per_hour'
'median_views_per_hour'
'avg_hours_with_views_per_day'
'max_hours_with_views_per_day'
'median_hours_with_views_per_day'
'tot_participations'
'average_daily_participations'
'median_daily_participations'
'max_daily_participation'
'days_with_participations'
```

EDA – Missing Values & Outliers

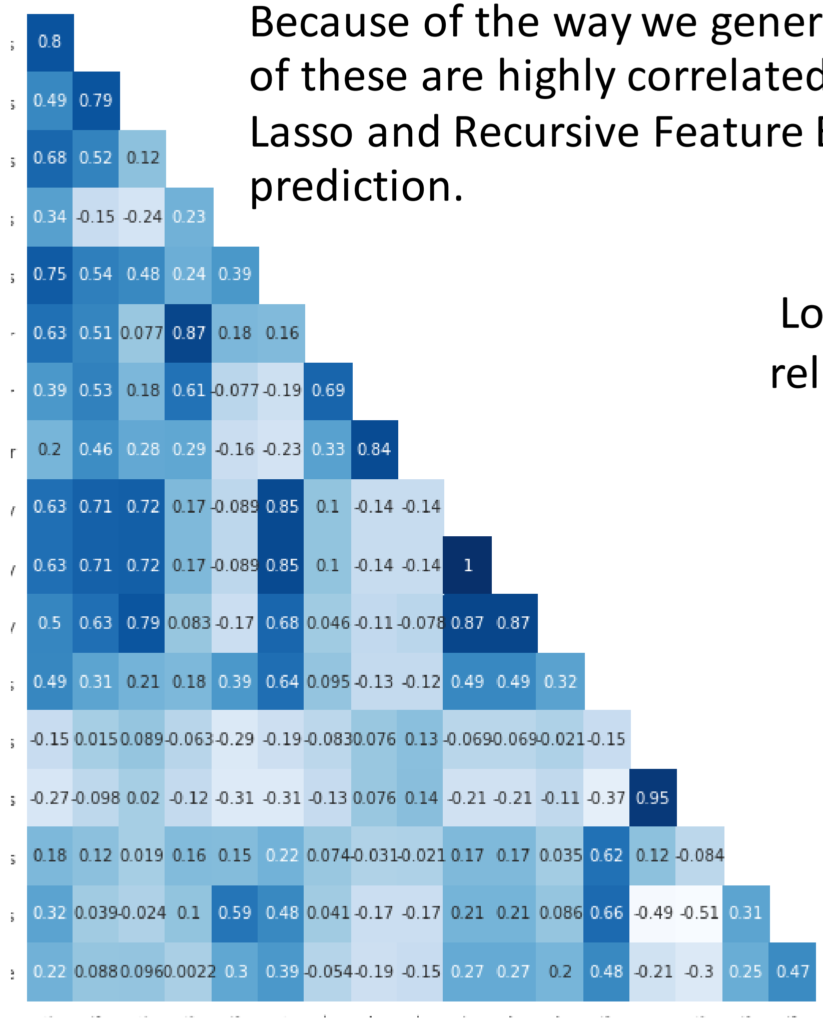
These creates a small but clean dataset, with only a few missing values that can be safely discarded.

The outliers that are identified represent real extreme values and thus should be preserved.



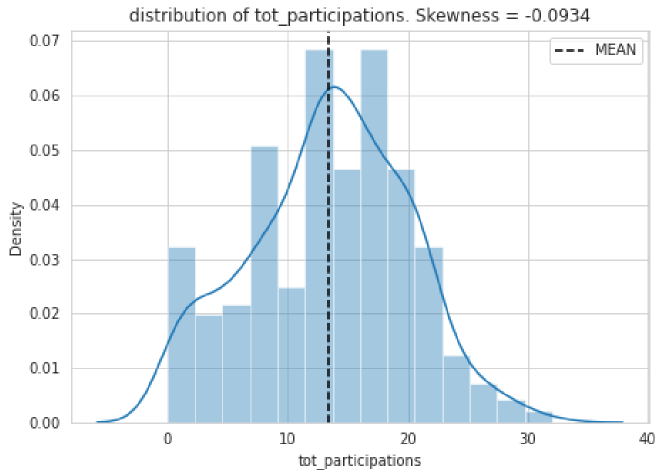


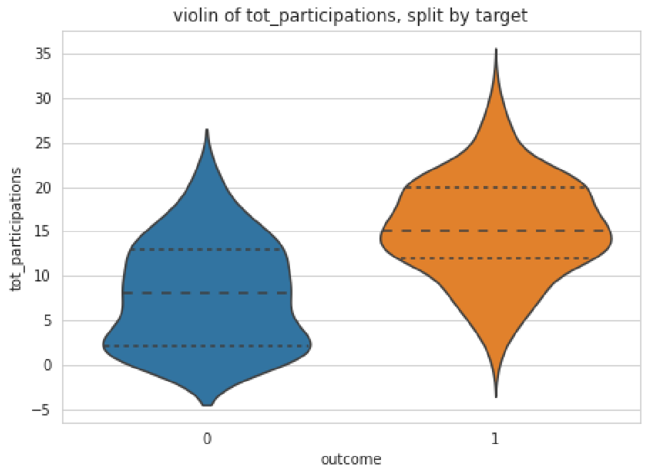
EDA – Correlation and Skewedness



Because of the way we generated ~16 features from just two time series it's no surprise that many of these are highly correlated. However instead of applying a correlation based filter we will use Lasso and Recursive Feature Elimination to identify the best features to preserve for the prediction.

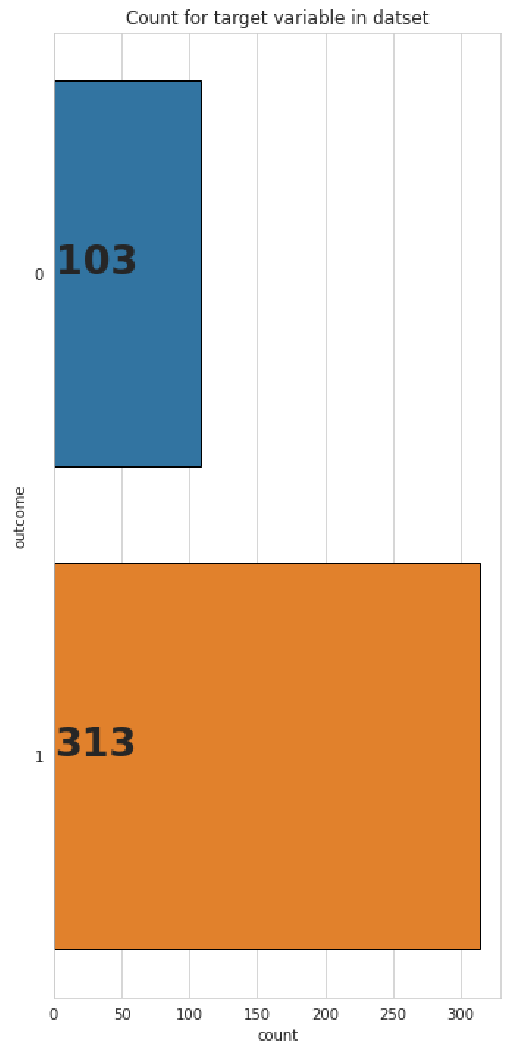
Looking at the distribution skewedness of the variable we see relevant differences between the two populations.

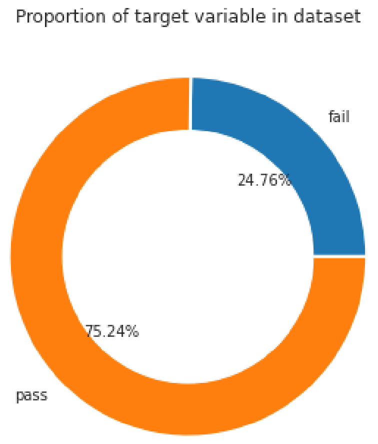




EDA – Outcome Variable

The class imbalance in the outcome variable is not a concern.



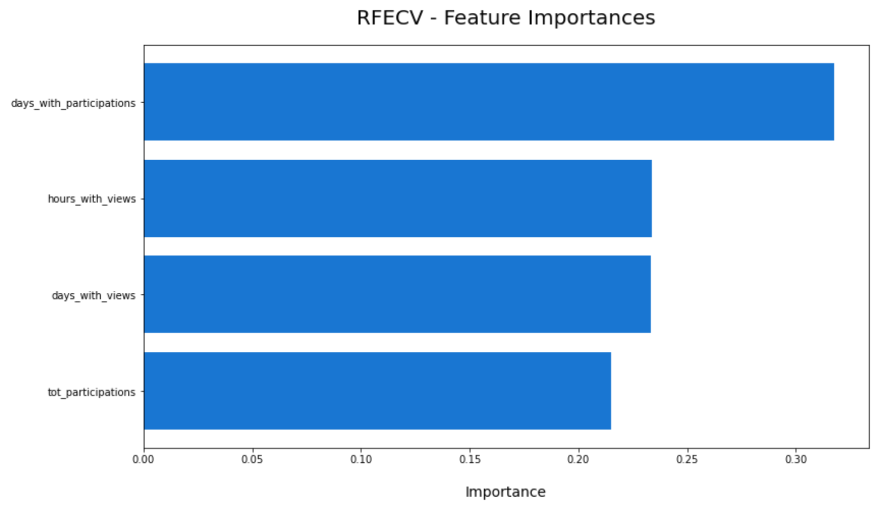


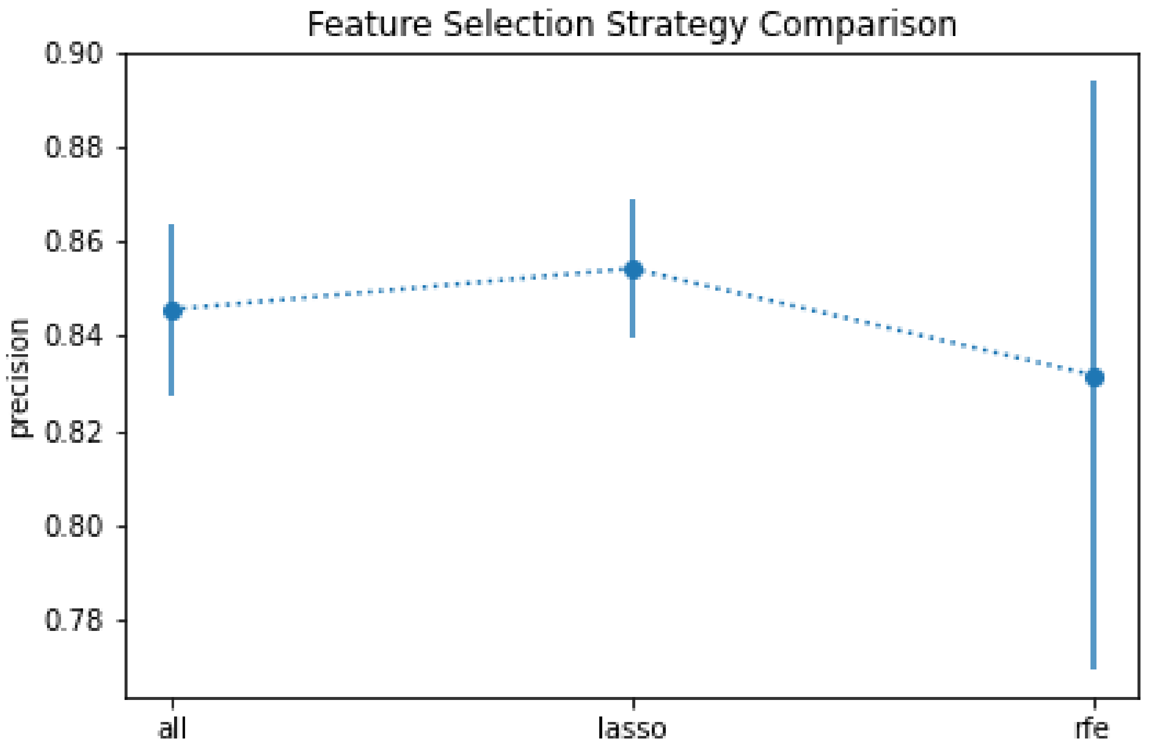
Feature Selection

Both Lasso and Recursive Feature Identification find that 4 variables are sufficient for the prediction, with minor differences.

There is however a clear advantage in the performance of the Lasso selected features.

```
Features Selected with  Lasso
- hours_with_views
- avg_views_per_hour
- tot_participations
- days_with_participations
```

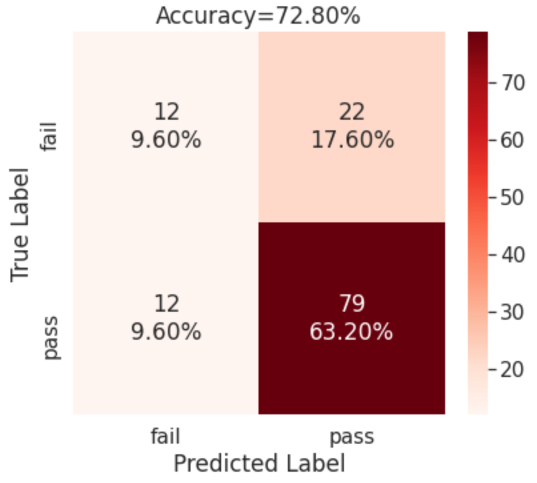




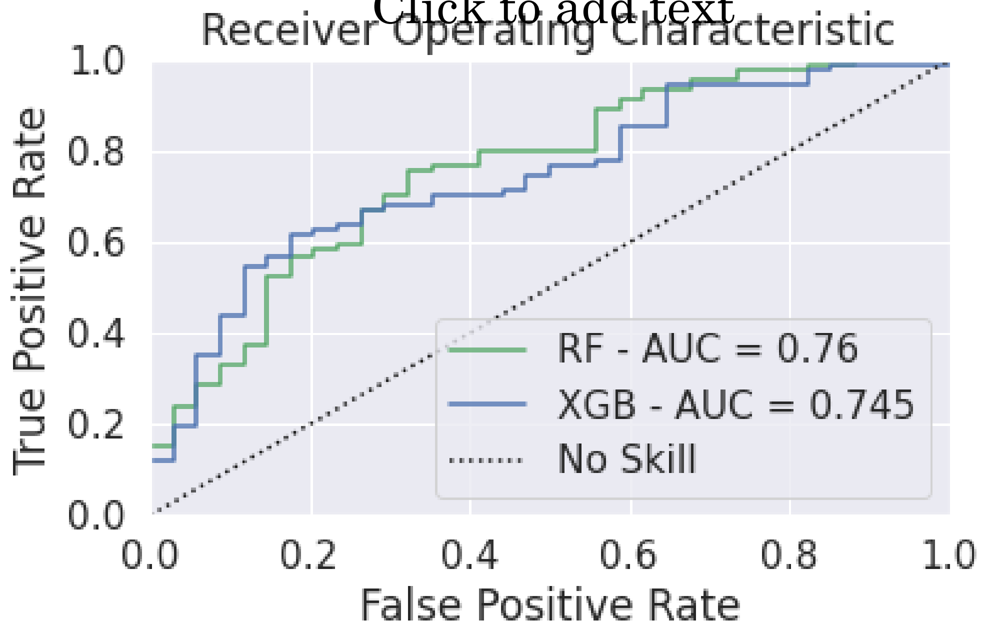
Modeling

We compare a Random Forest and a XGBoost Classifier to see which one results in better precision. We can also see that the Random Forest captures a larger area under the ROC Curve, and has fewer false positives

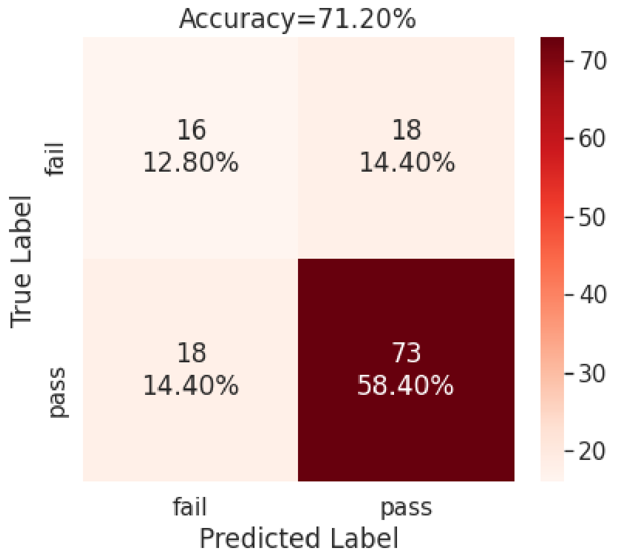
XGBoost



Click to add text



Random Forest



Model Selection

Based on these results this is the Random Forest is the model of choice

```
Filename : ../models/RandomForestClassifier__precision.pkl

Model type : <class 'sklearn.ensemble._forest.RandomForestClassifier'>

Model Parameters : OrderedDict([('class_weight', 'balanced'), ('max_depth', 3),
('min_samples_leaf', 4), ('min_samples_split', 4), ('n_estimators', 316)])

CV Score : 0.9047453703703704

Test Score: 0.8352941176470589
```

With the following predictor variables:

```
- hours_with_views
- avg_views_per_hour
- tot_participations
- days_with_participations
```