Assignment 2 | Building Random Forest Classifiers

Lars Michael Alexander Tallqvist | 1901050

https://github.com/AlexanderTallqvist/AA-DATA-SCIENCE

Introduction

This assignment discusses the process of using a random forest classifier to predict students' final grade in an online course. We were given a set of data that we were to analyze and use to build a classifier to predict student grades based on certain features from the data set. In short, these are the tasks that were to be completed for this assignment:

1. Process the given data and decide on features to keep.
2. Divide the dataset and use a portion of it for training.
3. Evaluate the performance of the models that you trained.

Data processing

The data the we received contained a total of 107 rows of student data based on an old course. There was a total of 48 columns of various information. Nine (9) of these columns contained data related to assignment grades, and 36 were log entries collected of various activities done on a courses Moodle page.

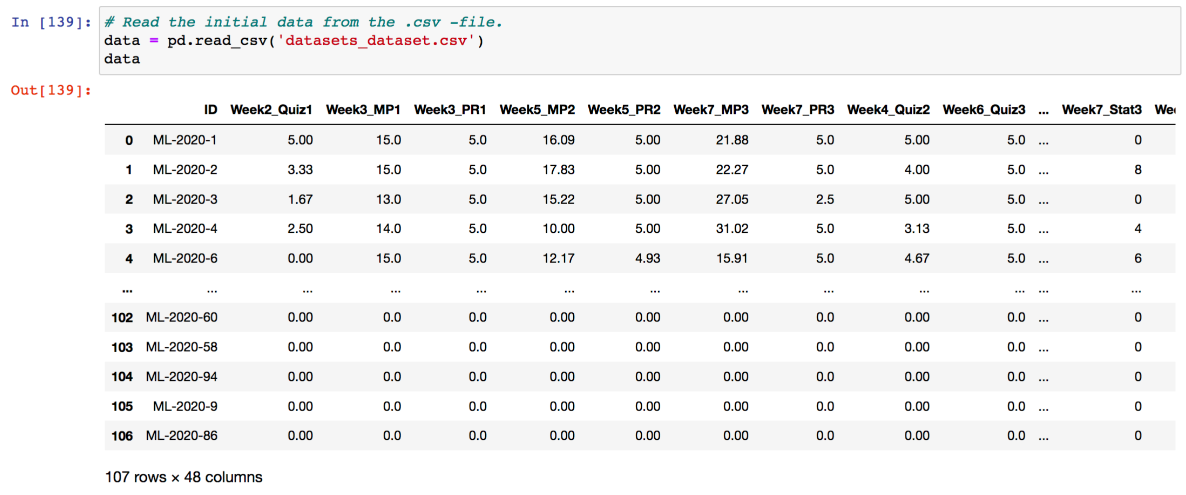
- **Grades:** 9 columns contained data related to grades.
- **Status0:** 9 columns contained data related to loge entries regarding the course itself, lectures and the course content: *(Course module viewed, Course viewed, Course activity completion updated, Course module instance list viewed, Content page viewed, Lesson started, Lesson resumed, Lesson restarted, Lesson ende).*
- **Status1:** 9 columns contained data related to loge entries regarding the course assignments and quizzes: *(Quiz attempt reviewed, Quiz attempt submitted, Quiz attempt summary viewed, Quiz attempt viewed, Quiz attempt started, Question answered, Question viewed, Submission re-assessed, Submission assessed, Submission updated, Submission created, Submission viewed)*

- **Status2**: 9 columns contained data related to loge entries regarding the grades and report views: *(Grade user report viewed, Grade overview report viewed, User graded, Grade deleted, User profile viewed, Recent activity viewed, User report viewed, Course user report viewed, Outline report viewed).*
- **Status3**: 9 columns contained data related to loge entries regarding the forum activities: *(Post updated, Post created, Discussion created, Some content has been posted, Discussion viewed).*

After looking through the data and making sure that all the columns and rows had entries in them, it was time to decide on the features. I ended up using three different sets of features, and then analyzed their results against each other. The first feature set contained all the grade columns, the second feature set contained all the grade columns and the columns from Status0 and Status1, and the third feature set contained all the grade columns (excluding the final/total grade from week 8) and the columns from Status0 and Status1. I chose to use the columns from Status0 and Status1 as features because they seemed most relevant to me, and also contained the most information from all the "Status" columns. The feature sets are described below:

1. **Feature set 1:** The 10 columns containing grades (including the total score). A total of 10 features: *('Week2_Quiz1', 'Week3_MP1', 'Week3_PR1', 'Week5_MP2', 'Week5_PR2', 'Week7_MP3', 'Week7_PR3', 'Week6_Quiz3', 'Week4_Quiz2', 'Week8_Total').*
2. **Feature set 2:** The 10 columns containing grades (including the total score) and the columns from Status0 and Status1. A total of 28 features: *('Week2_Quiz1', 'Week3_MP1', 'Week3_PR1', 'Week5_MP2', 'Week5_PR2', 'Week7_MP3', 'Week7_PR3', 'Week6_Quiz3', 'Week4_Quiz2', 'Week8_Total', 'Week1_Stat0','Week1_Stat1', 'Week2_Stat0','Week2_Stat1', 'Week3_Stat0','Week3_Stat1', 'Week4_Stat0','Week4_Stat1', 'Week5_Stat0','Week5_Stat1', 'Week6_Stat0','Week6_Stat1', 'Week7_Stat0','Week7_Stat1', 'Week8_Stat0','Week8_Stat1', 'Week9_Stat0','Week9_Stat1').*

3. **Feature set 3:** The 9 columns containing grades (**NOT including the total score**) and the columns from Status0 and Status1. A total of 27 features.



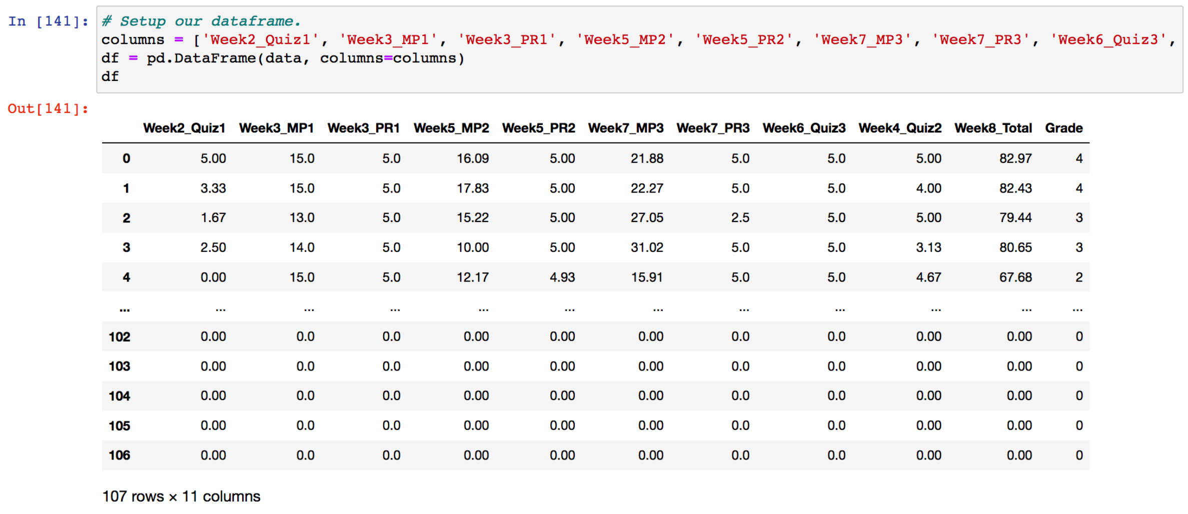
```
In [139]: # Read the initial data from the .csv -file.
          data = pd.read_csv('datasets_dataset.csv')
          data
```

Out[139]:

	ID	Week2_Quiz1	Week3_MP1	Week3_PR1	Week5_MP2	Week5_PR2	Week7_MP3	Week7_PR3	Week4_Quiz2	Week6_Quiz3	...	Week7_Stat3	We
0	ML-2020-1	5.00	15.0	5.0	16.09	5.00	21.88	5.0	5.00	5.0	...	0	
1	ML-2020-2	3.33	15.0	5.0	17.83	5.00	22.27	5.0	4.00	5.0	...	8	
2	ML-2020-3	1.67	13.0	5.0	15.22	5.00	27.05	2.5	5.00	5.0	...	0	
3	ML-2020-4	2.50	14.0	5.0	10.00	5.00	31.02	5.0	3.13	5.0	...	4	
4	ML-2020-6	0.00	15.0	5.0	12.17	4.93	15.91	5.0	4.67	5.0	...	6	
...	...	...	...	...	...	...	...	...	...	...	...	...	
102	ML-2020-60	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.0	...	0	
103	ML-2020-58	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.0	...	0	
104	ML-2020-94	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.0	...	0	
105	ML-2020-9	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.0	...	0	
106	ML-2020-86	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.0	...	0	

107 rows × 48 columns

Image1: Reading data from the provided .csv file.



```
In [141]: # Setup our dataframe.
          columns = ['Week2_Quiz1', 'Week3_MP1', 'Week3_PR1', 'Week5_MP2', 'Week5_PR2', 'Week7_MP3', 'Week7_PR3', 'Week6_Quiz3',
          df = pd.DataFrame(data, columns=columns)
          df
```

Out[141]:

	Week2_Quiz1	Week3_MP1	Week3_PR1	Week5_MP2	Week5_PR2	Week7_MP3	Week7_PR3	Week6_Quiz3	Week4_Quiz2	Week8_Total	Grade
0	5.00	15.0	5.0	16.09	5.00	21.88	5.0	5.0	5.00	82.97	4
1	3.33	15.0	5.0	17.83	5.00	22.27	5.0	5.0	4.00	82.43	4
2	1.67	13.0	5.0	15.22	5.00	27.05	2.5	5.0	5.00	79.44	3
3	2.50	14.0	5.0	10.00	5.00	31.02	5.0	5.0	3.13	80.65	3
4	0.00	15.0	5.0	12.17	4.93	15.91	5.0	5.0	4.67	67.68	2
...	...	...	...	...	...	...	...	...	...	...	...
102	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.00	0.00	0
103	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.00	0.00	0
104	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.00	0.00	0
105	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.00	0.00	0
106	0.00	0.0	0.0	0.00	0.00	0.00	0.0	0.0	0.00	0.00	0

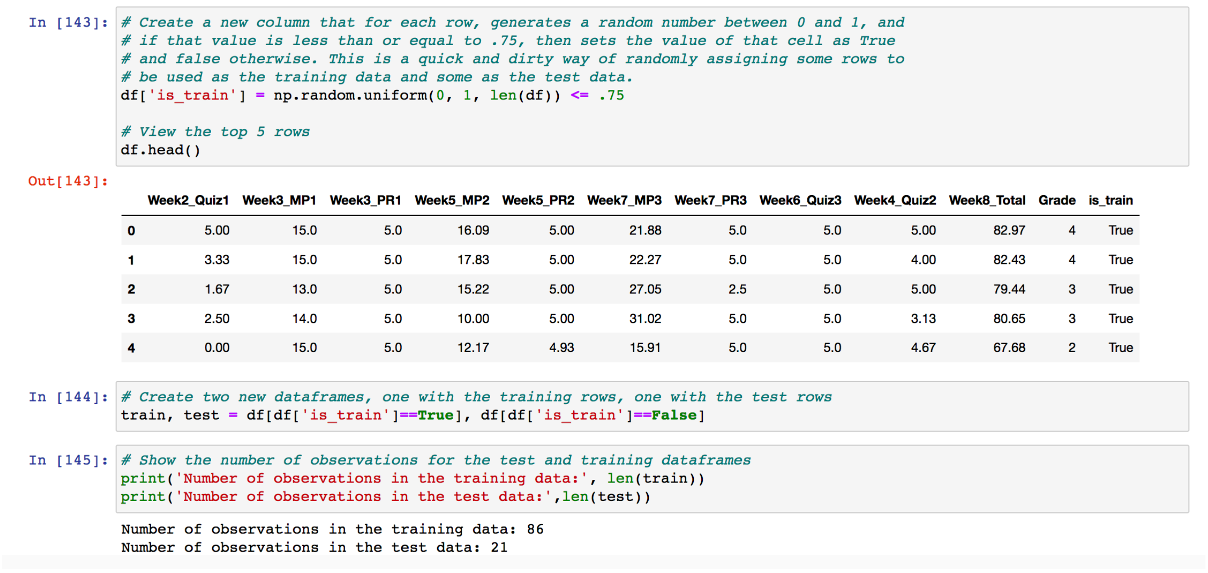
107 rows × 11 columns

Image 2: Narrowing down the data from feature set 1.

Modelling

After the features had been selected, it was time to start training the models. I chose to insert a new column in the data set called "is_training". The is_training column would randomly be set to either true or false for each row, keeping in mind that the probability of the value being true was set to 75%. This way I would get a split of about 75% to 25%, where the bigger portion was used for training the models. The training/testing split that I ended up getting for each feature set was the following:

- **Feature set 1:** 86 rows for training, 21 for testing.
- **Feature set 2:** 77 rows for training, 30 for testing.
- **Feature set 3:** 82 rows for training, 25 for testing.



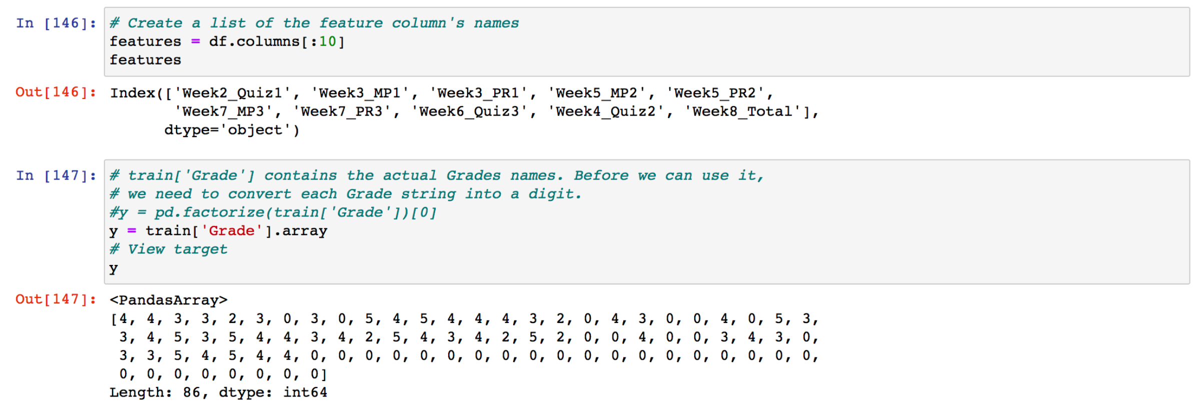
```
In [143]: # Create a new column that for each row, generates a random number between 0 and 1, and
          # if that value is less than or equal to .75, then sets the value of that cell as True
          # and false otherwise. This is a quick and dirty way of randomly assigning some rows to
          # be used as the training data and some as the test data.
          df['is_train'] = np.random.uniform(0, 1, len(df)) <= .75

          # View the top 5 rows
          df.head()
```

Out[143]:

	Week2_Quiz1	Week3_MP1	Week3_PR1	Week5_MP2	Week5_PR2	Week7_MP3	Week7_PR3	Week6_Quiz3	Week4_Quiz2	Week8_Total	Grade	is_train
0	5.00	15.0	5.0	16.09	5.00	21.88	5.0	5.0	5.00	82.97	4	True
1	3.33	15.0	5.0	17.83	5.00	22.27	5.0	5.0	4.00	82.43	4	True
2	1.67	13.0	5.0	15.22	5.00	27.05	2.5	5.0	5.00	79.44	3	True
3	2.50	14.0	5.0	10.00	5.00	31.02	5.0	5.0	3.13	80.65	3	True
4	0.00	15.0	5.0	12.17	4.93	15.91	5.0	5.0	4.67	67.68	2	True

```
In [144]: # Create two new dataframes, one with the training rows, one with the test rows
          train, test = df[df['is_train']==True], df[df['is_train']==False]

In [145]: # Show the number of observations for the test and training dataframes
          print('Number of observations in the training data:', len(train))
          print('Number of observations in the test data:',len(test))

          Number of observations in the training data: 86
          Number of observations in the test data: 21
```

Image 3: Inserting the is_training column in the dataset for feature set 1, and splitting the data into training and testing data.



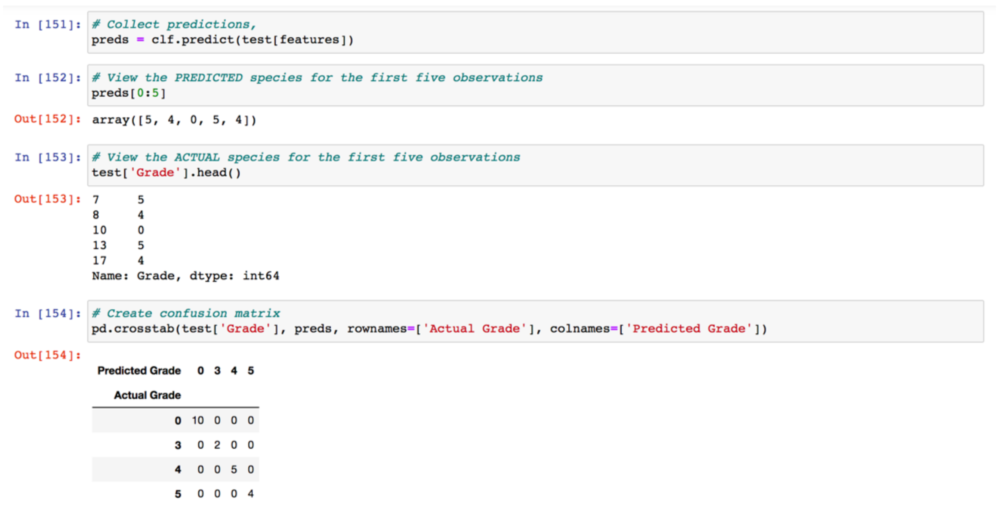
```
In [146]: # Create a list of the feature column's names
          features = df.columns[:10]
          features

Out[146]: Index(['Week2_Quiz1', 'Week3_MP1', 'Week3_PR1', 'Week5_MP2', 'Week5_PR2',
                 'Week7_MP3', 'Week7_PR3', 'Week6_Quiz3', 'Week4_Quiz2', 'Week8_Total'],
                dtype='object')

In [147]: # train['Grade'] contains the actual Grades names. Before we can use it,
          # we need to convert each Grade string into a digit.
          #y = pd.factorize(train['Grade'])[0]
          y = train['Grade'].array
          # View target
          y

Out[147]: <PandasArray>
          [4, 4, 3, 3, 2, 3, 0, 3, 0, 5, 4, 5, 4, 4, 4, 3, 2, 0, 4, 3, 0, 0, 4, 0, 5, 3,
           3, 4, 5, 3, 5, 4, 4, 3, 4, 2, 5, 4, 3, 4, 2, 5, 2, 0, 0, 4, 0, 0, 3, 4, 3, 0,
           3, 3, 5, 4, 5, 4, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0]
          Length: 86, dtype: int64
```

Image 4: Setting the features for feature set 1 and setting the Y/target value.

Results

Creating Random Forest Classifiers of our models and running them through the training data produced the following results:

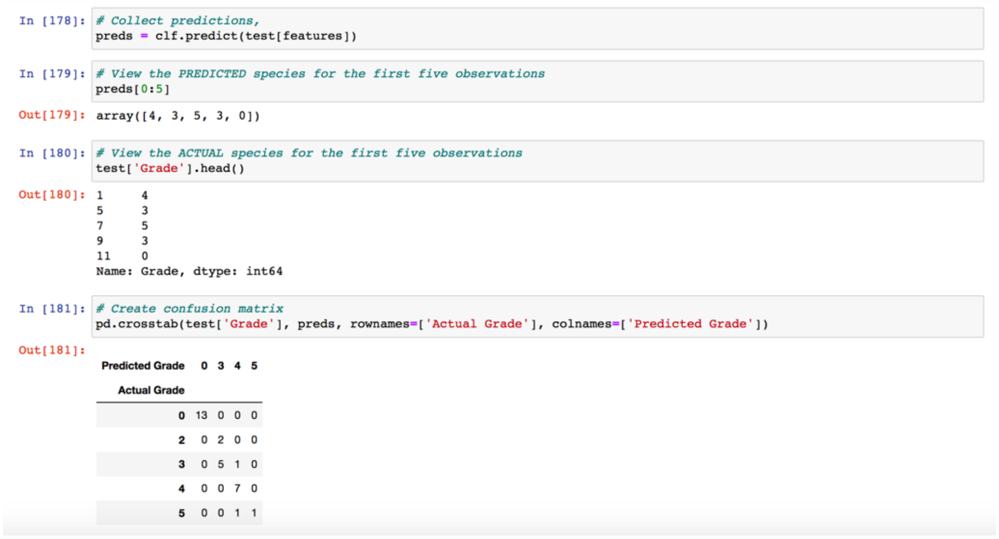
- **Feature set 1:** 100% of the predictions were correct (21/21).
- **Feature set 2:** 86,66% of the predictions were correct (26/30).
- **Feature set 3:** 76% of the predictions were correct (19/25).



```
In [151]: # Collect predictions,
          preds = clf.predict(test[features])

In [152]: # View the PREDICTED species for the first five observations
          preds[0:5]

Out[152]: array([5, 4, 0, 5, 4])

In [153]: # View the ACTUAL species for the first five observations
          test['Grade'].head()

Out[153]: 7     5
          8     4
          10    0
          13    5
          17    4
          Name: Grade, dtype: int64

In [154]: # Create confusion matrix
          pd.crosstab(test['Grade'], preds, rownames=['Actual Grade'], colnames=['Predicted Grade'])

Out[154]:
```

Predicted Grade	0	3	4	5
Actual Grade				
0	10	0	0	0
3	0	2	0	0
4	0	0	5	0
5	0	0	0	4

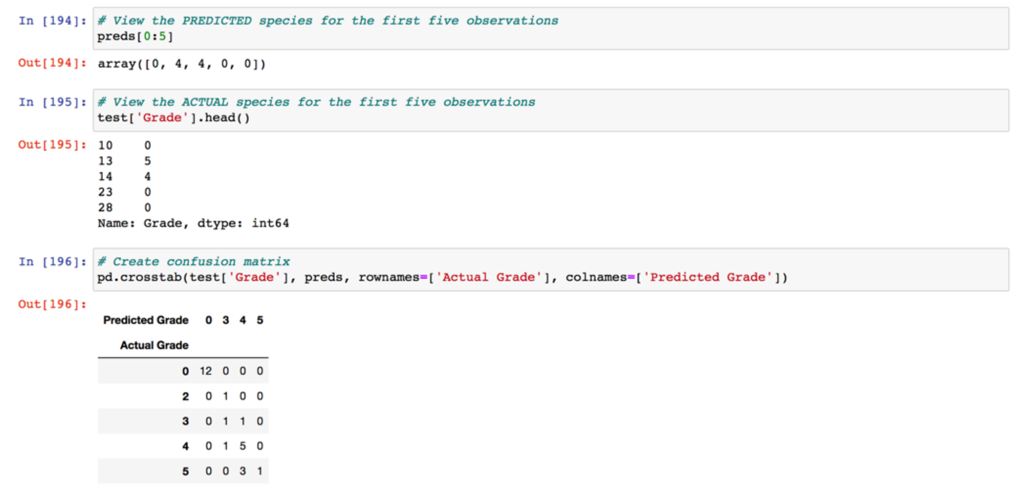
Image 5: Results for **features set 1** presented in a confusion matrix.



```
In [178]: # Collect predictions,
          preds = clf.predict(test[features])

In [179]: # View the PREDICTED species for the first five observations
          preds[0:5]

Out[179]: array([4, 3, 5, 3, 0])

In [180]: # View the ACTUAL species for the first five observations
          test['Grade'].head()

Out[180]: 1     4
          5     3
          7     5
          9     3
          11    0
          Name: Grade, dtype: int64

In [181]: # Create confusion matrix
          pd.crosstab(test['Grade'], preds, rownames=['Actual Grade'], colnames=['Predicted Grade'])

Out[181]:
```

Predicted Grade	0	3	4	5
Actual Grade				
0	13	0	0	0
2	0	2	0	0
3	0	5	1	0
4	0	0	7	0
5	0	0	1	1

Image 6: Results for **features set 2** presented in a confusion matrix.

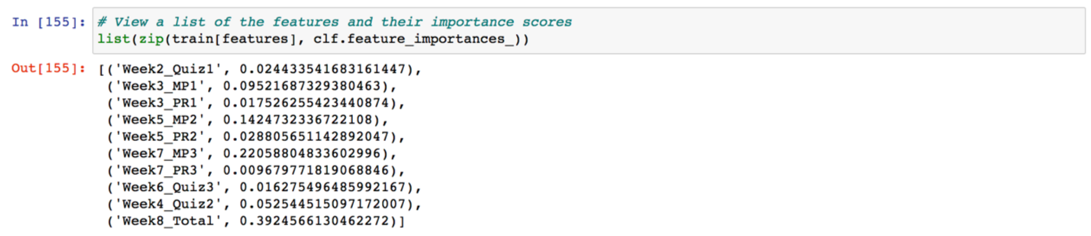


```
In [194]: # View the PREDICTED species for the first five observations
          preds[0:5]

Out[194]: array([0, 4, 4, 0, 0])

In [195]: # View the ACTUAL species for the first five observations
          test['Grade'].head()

Out[195]: 10    0
          13    5
          14    4
          23    0
          28    0
          Name: Grade, dtype: int64

In [196]: # Create confusion matrix
          pd.crosstab(test['Grade'], preds, rownames=['Actual Grade'], colnames=['Predicted Grade'])

Out[196]:
```

Predicted Grade	0	3	4	5
Actual Grade				
0	12	0	0	0
2	0	1	0	0
3	0	1	1	0
4	0	1	5	0
5	0	0	3	1

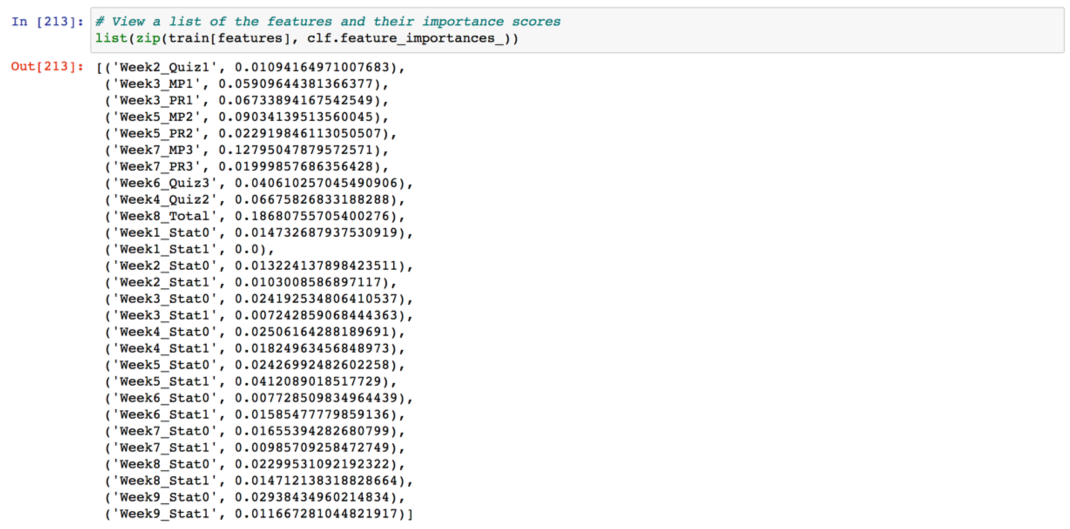
Image 7: Results for **features set 3** presented in a confusion matrix.

The most important features from each feature set were the grades related to the mini projects, and the total score if it was included. The results and the importance of the features is presented below:



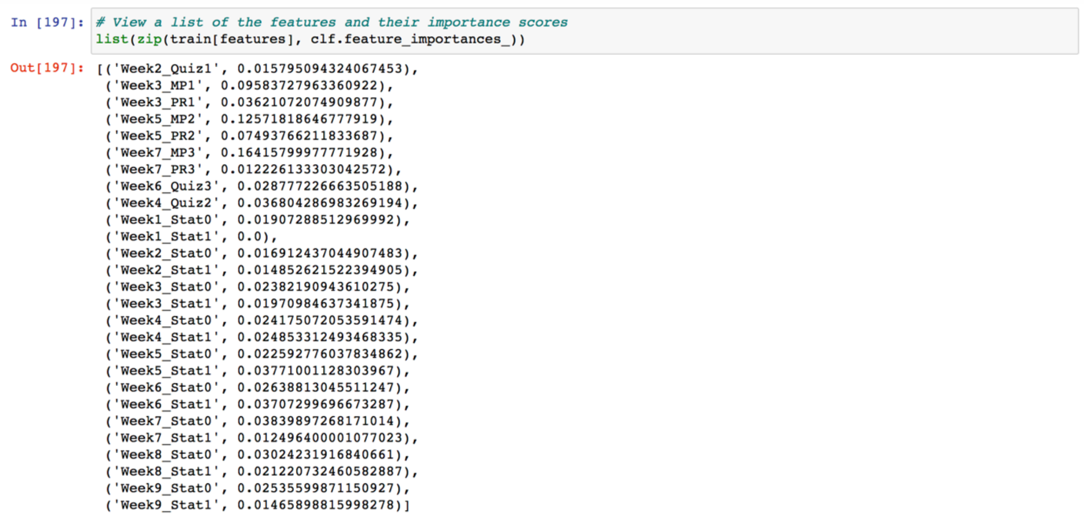
```
In [155]: # View a list of the features and their importance scores
          list(zip(train[features], clf.feature_importances_))

Out[155]: [('Week2_Quiz1', 0.024433541683161447),
           ('Week3_MP1', 0.09521687329380463),
           ('Week3_PR1', 0.017526255423440874),
           ('Week5_MP2', 0.1424732336722108),
           ('Week5_PR2', 0.028805651142892047),
           ('Week7_MP3', 0.22058804833602996),
           ('Week7_PR3', 0.009679771819068846),
           ('Week6_Quiz3', 0.016275496485992167),
           ('Week4_Quiz2', 0.052544515097172007),
           ('Week8_Total', 0.3924566130462272)]
```

Image 8: Most important features for **feature set 1**: *Week8_total, Week7_MP3, Week5_MP2*.



```
In [213]: # View a list of the features and their importance scores
          list(zip(train[features], clf.feature_importances_))

Out[213]: [('Week2_Quiz1', 0.01094164971007683),
           ('Week3_MP1', 0.05909644381366377),
           ('Week3_PR1', 0.06733894167542549),
           ('Week5_MP2', 0.09034139513560045),
           ('Week5_PR2', 0.022919846113050507),
           ('Week7_MP3', 0.12795047879572571),
           ('Week7_PR3', 0.01999857686356428),
           ('Week6_Quiz3', 0.040610257045490906),
           ('Week4_Quiz2', 0.06675826833188288),
           ('Week8_Total', 0.18680755705400276),
           ('Week1_Stat0', 0.014732687937530919),
           ('Week1_Stat1', 0.0),
           ('Week2_Stat0', 0.013224137898423511),
           ('Week2_Stat1', 0.0103008586897117),
           ('Week3_Stat0', 0.024192534806410537),
           ('Week3_Stat1', 0.007242859068444363),
           ('Week4_Stat0', 0.02506164288189691),
           ('Week4_Stat1', 0.01824963456848973),
           ('Week5_Stat0', 0.02426992482602258),
           ('Week5_Stat1', 0.0412089018517729),
           ('Week6_Stat0', 0.007728509834964439),
           ('Week6_Stat1', 0.01585477779859136),
           ('Week7_Stat0', 0.01655394282680799),
           ('Week7_Stat1', 0.00985709258472749),
           ('Week8_Stat0', 0.02299531092192322),
           ('Week8_Stat1', 0.014712138318828664),
           ('Week9_Stat0', 0.02938434960214834),
           ('Week9_Stat1', 0.011667281044821917)]
```

Image 9: Most important features for **feature set 2**: Week8_total, Week7_MP3, Week5_MP2.



```
In [197]: # View a list of the features and their importance scores
          list(zip(train[features], clf.feature_importances_))

Out[197]: [('Week2_Quiz1', 0.015795094324067453),
           ('Week3_MP1', 0.09583727963360922),
           ('Week3_PR1', 0.03621072074909877),
           ('Week5_MP2', 0.12571818646777919),
           ('Week5_PR2', 0.07493766211833687),
           ('Week7_MP3', 0.16415799977771928),
           ('Week7_PR3', 0.012226133303042572),
           ('Week6_Quiz3', 0.028777226663505188),
           ('Week4_Quiz2', 0.036804286983269194),
           ('Week1_Stat0', 0.01907288512969992),
           ('Week1_Stat1', 0.0),
           ('Week2_Stat0', 0.016912437044907483),
           ('Week2_Stat1', 0.014852621522394905),
           ('Week3_Stat0', 0.02382190943610275),
           ('Week3_Stat1', 0.01970984637341875),
           ('Week4_Stat0', 0.024175072053591474),
           ('Week4_Stat1', 0.024853312493468335),
           ('Week5_Stat0', 0.022592776037834862),
           ('Week5_Stat1', 0.03771001128303967),
           ('Week6_Stat0', 0.02638813045511247),
           ('Week6_Stat1', 0.03707299696673287),
           ('Week7_Stat0', 0.03839897268171014),
           ('Week7_Stat1', 0.012496400001077023),
           ('Week8_Stat0', 0.03024231916840661),
           ('Week8_Stat1', 0.021220732460582887),
           ('Week9_Stat0', 0.02535599871150927),
           ('Week9_Stat1', 0.01465898815998278)]
```

Image 10: Most important features for **feature set 3**: *Week7_MP3, Week5_MP2, Weel3_MP1*.

Conclusion

Using Random Forest Classifiers can be a powerful tool when used correctly with the correct features. The models that I ended up creating turned out to be relatively accurate, even when the final score was excluded from the feature set. I had never worked with Random Forest Classifier before, so I can definitely say that I've learned a lot. All in all this assignment was both fun and challenging, and I look forward to applying my newly acquired knowledge in the future.