Project Two: Classification

Zoe Markovits
Data Mining - CSC 84040

Dataset and Goals:

- Kaggle’s Students’ Academic Performance Dataset
- Our Goal: Can we predict and classify a student's academic performance as either performing high, medium or low?
- Features broken up into three categories: demographic features, academic background features, and behavioral features

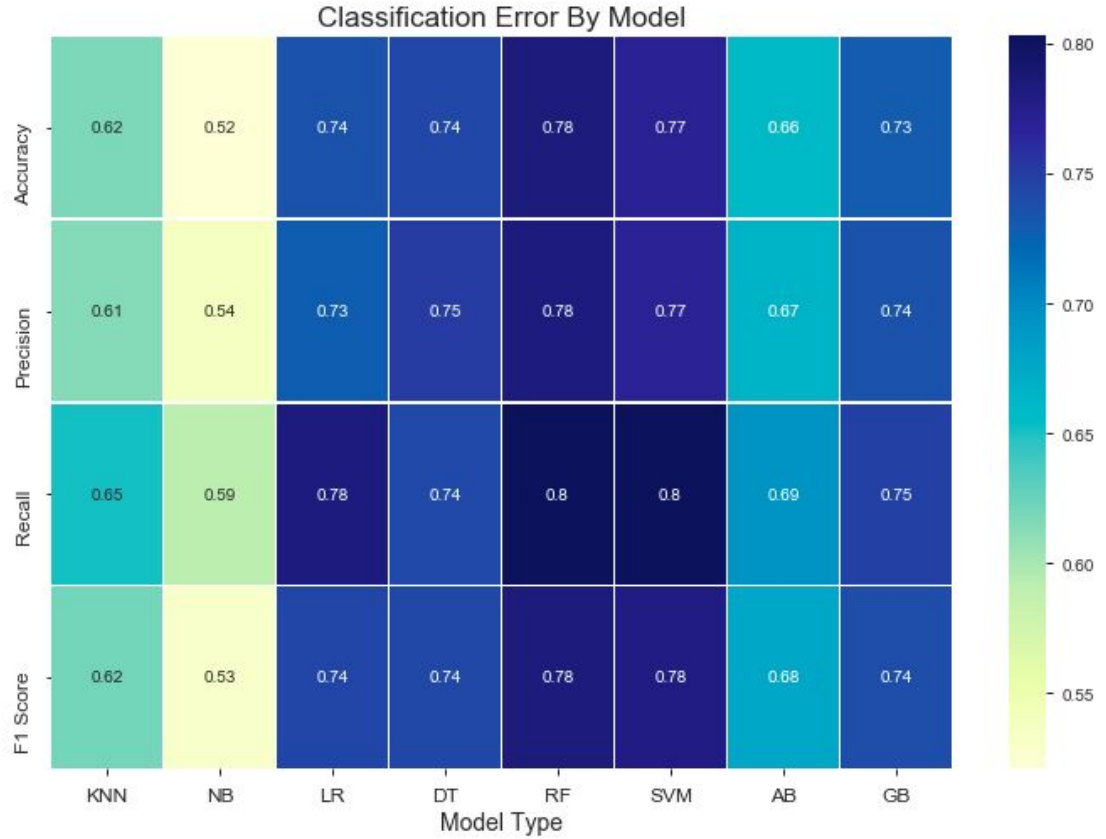
Dataset and Goals:

	Performance	Gender	Nationality	Place_of_Birth	Education_Level	Grade_Level	Section_ID	Topic	Semester
0	Medium	Male	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First
1	Medium	Male	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First
2	Low	Male	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First
3	Low	Male	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First
4	Medium	Male	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First
5	Medium	Female	Kuwait	Kuwait	Elementary_School	Grade_Four	A	IT	First

Parent	Raised_Hand	Visited_Resources	Viewed_Announcements	Discussion_Groups	Parent_Answered_Survey	Parent_School_Satisfaction	Absences
Father	15	16	2	20	Yes	Good	Under_Seven
Father	20	20	3	25	Yes	Good	Under_Seven
Father	10	7	0	30	No	Bad	Over_Seven
Father	30	25	5	35	No	Bad	Over_Seven
Father	40	50	12	50	No	Bad	Over_Seven

Baseline Models:

After cleaning and preprocessing our data we run baseline classification models with Naive Bayes, K-Nearest Neighbors, Support Vector Machine, Logistic Regression, Decision Tree, Random Forest, AdaBoost, and GradientBoost



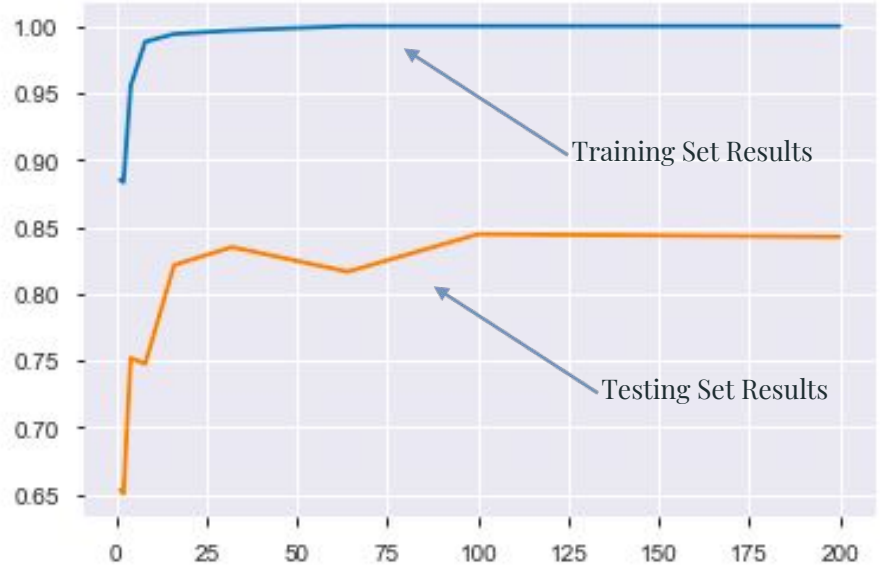
We can see that our two best models are Random Forest and SVM and that our strongest evaluation metric is recall

Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

First we look at the number of n_estimators, or the number of trees we're using in the forest.

This shows us that after 32 trees our score tends to decrease and/or flatten out.



Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

Next, we look at the max-depth of each tree.

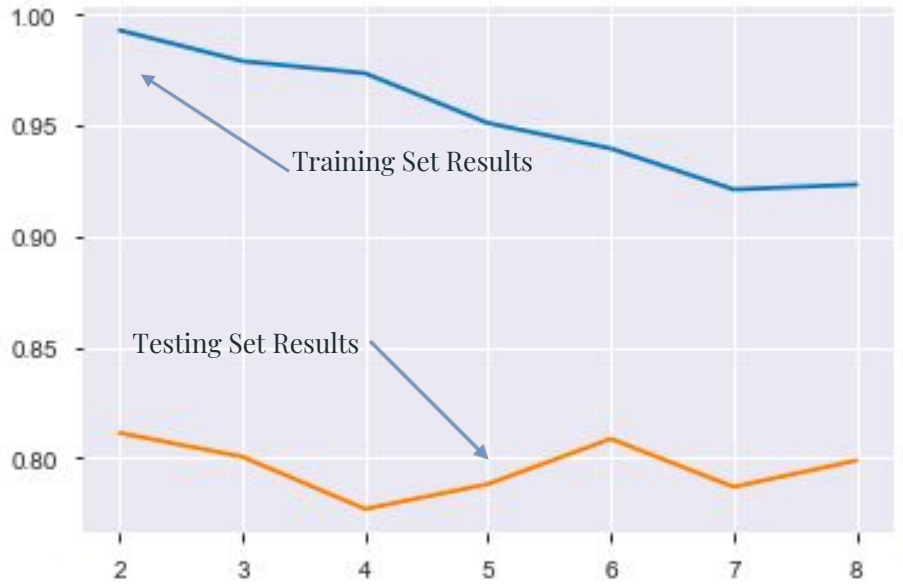
We can see that as max depth increases our model overfits, thus we settle on a max depth of 6.

Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

Next we look at our min_sample_split, or the minimum number of samples required to split an internal node

Our testing set has its highest score when the minimum number of samples is 6

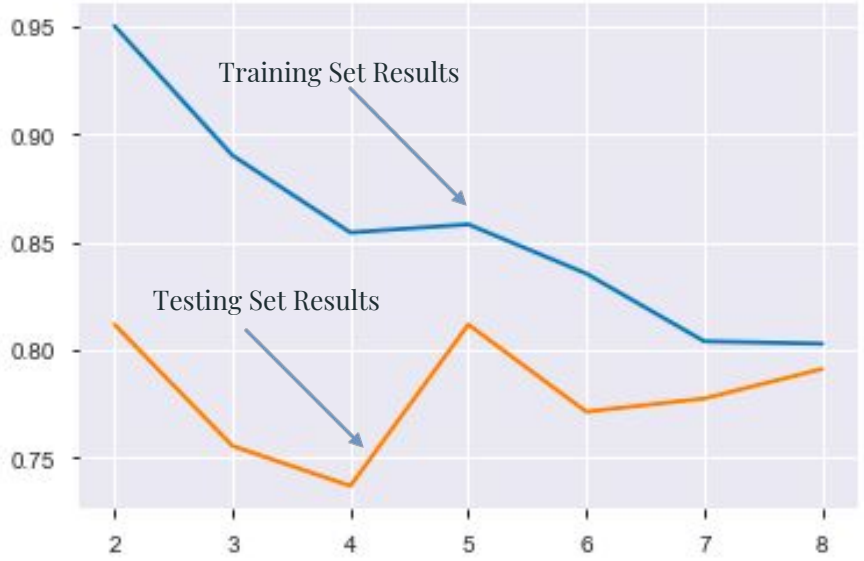


Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

Finally we look at our min_samples_leaf, or the minimum number of samples required to be at a leaf node

Our testing set has its highest score when the minimum number of samples is 5



Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

```
RandomForest_tuned = RandomForestClassifier(n_jobs=-1,n_estimators=32,max_depth=6,min_samples_split=6,min_samples_leaf
=5)
RandomForest_tuned.fit(X_train,y_train)
RandomForest_predict = RandomForest_tuned.predict(X_test)

RandomForest_accuracy = accuracy_score(y_test, RandomForest_predict)
RandomForest_precision = precision_score(y_test, RandomForest_predict, average='macro')
RandomForest_recall = recall_score(y_test, RandomForest_predict, average='macro')
RandomForest_f1 = f1_score(y_test, RandomForest_predict, average='macro')

print(RandomForest_accuracy)
print(RandomForest_precision)
print(RandomForest_recall)
print(RandomForest_f1)
```

```
0.8194444444444444
0.820705227298752
0.8407730985370737
0.8254515599343186
```

We run our random forest model again with our newly tuned parameters.

Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

```
print("Confusion Matrix:\n", metrics.confusion_matrix(y_test, RandomForest_predict))
```

```
Confusion Matrix:
 [[36  0 13]
 [ 0 25  1]
 [ 5  7 57]]
```

```
print("Classification Report:\n", metrics.classification_report(y_test, RandomForest_predict))
```

```
Classification Report:
              precision    recall  f1-score   support

        High       0.88      0.73      0.80        49
         Low       0.78      0.96      0.86        26
      Medium       0.80      0.83      0.81        69

   micro avg       0.82      0.82      0.82       144
   macro avg       0.82      0.84      0.83       144
weighted avg       0.82      0.82      0.82       144
```

Optimized Random Forest:

Random Forest is our strongest model, we try to optimize it by tuning its hyperparameters.

```
features= zip(list(RandomForest.feature_importances_), X.columns)
sorted(features, key=lambda x: x[0], reverse=True)
```

```
[(0.12387720262583794, 'Raised_Hand'),
 (0.11270851698738142, 'Visited_Resources'),
 (0.10015996542557l4, 'Over_Seven'),
 (0.07856528125238924, 'Viewed_Announcements'),
 (0.06546465885962419, 'Discussion_Groups'),
 (0.062720407286610l6, 'Under_Seven'),
```

Strongest features are how many times a student raised their hand, visited course resources, viewed announcements, participated in discussion groups, and if they were absent over seven or under seven times so we can see that the behavioral features are most important

Conclusions

Thanks for listening!