Project 4: Analysis
Artificial Intelligence

Bibhash Mulmi

Hypothesis 1:

Who does better in Paradigm Course?

Sub data: Race No. 5

Input Variables:

1. Grade Fundamentals Course
2. Grade Systems Course
3. Grade Software Course

Goal Attribute: Grade in Paradigm Course

- Good Grades is greater than or equal to 3.0
- Bad Grades is less than 3.0

Data:

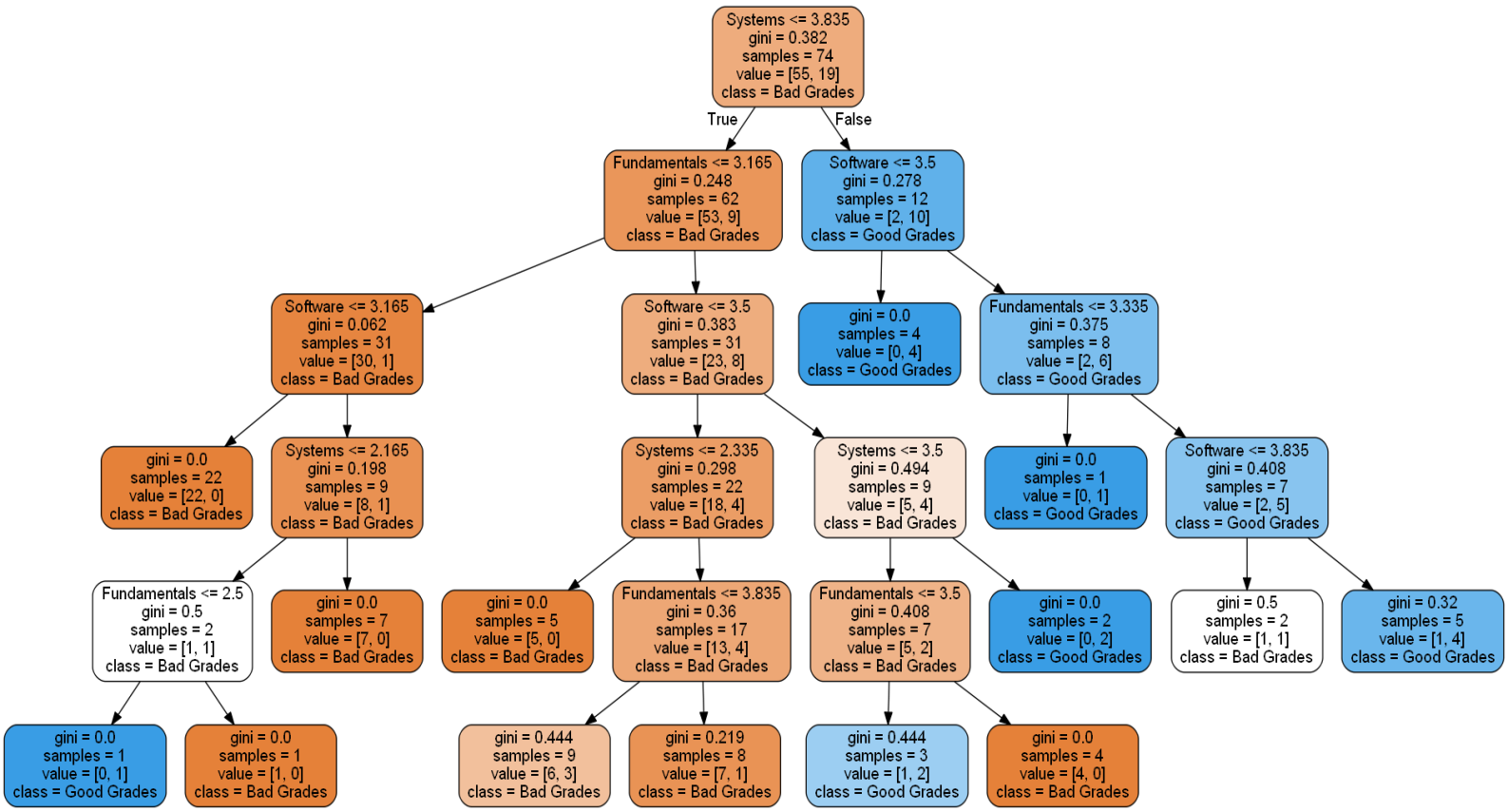
- Total Records = 106
- 70% training data = 74
- 30% testing data = 32

Feature Importance:

- Grade in Fundamentals Course: 0.24
- Grade in Systems Course: 0.52
- Grade in Software Course: 0.23

Accuracy: From Random Forest Classifier

- Test Data Accuracy = 81.25%
- Train Data Accuracy = 93.24%



Analysis:

- All students who received greater than 3.8 in Systems Course are predicted to get Good Grades regardless of their grade in Software or Fundamentals Course
- For students who received less than or equal to 3.8 in Systems Course:
  - With less than or equal to 3.1 in both Fundamentals and Software are predicted to get Bad Grades (1 exception: disregarded)
  - With greater than 3.1 in Fundamentals but less than or equal to 3.5 in Software, they are predicted to get Bad Grades
  - With Fundamentals course score and Systems course score within 3.1 to 3.5 and Software course score greater than 3.5 are predicted to get Good Grades
  - With greater than 3.5 in Software course and within 3.5 to 3.8 in Systems Course are predicted to get Good Grades

Hypothesis 2:

Who does better in Systems Course?

Sub data: Students who have taken Systems Course

Input Variables:

1. Grade in Intro Course
2. Grade in Follow Up Course
3. Grade in Fundamentals Course

Goal Attribute: Grade in Systems Course

- Good Grades is greater than or equal to 3.0
- Bad Grades is less than 3.0

Data Split:

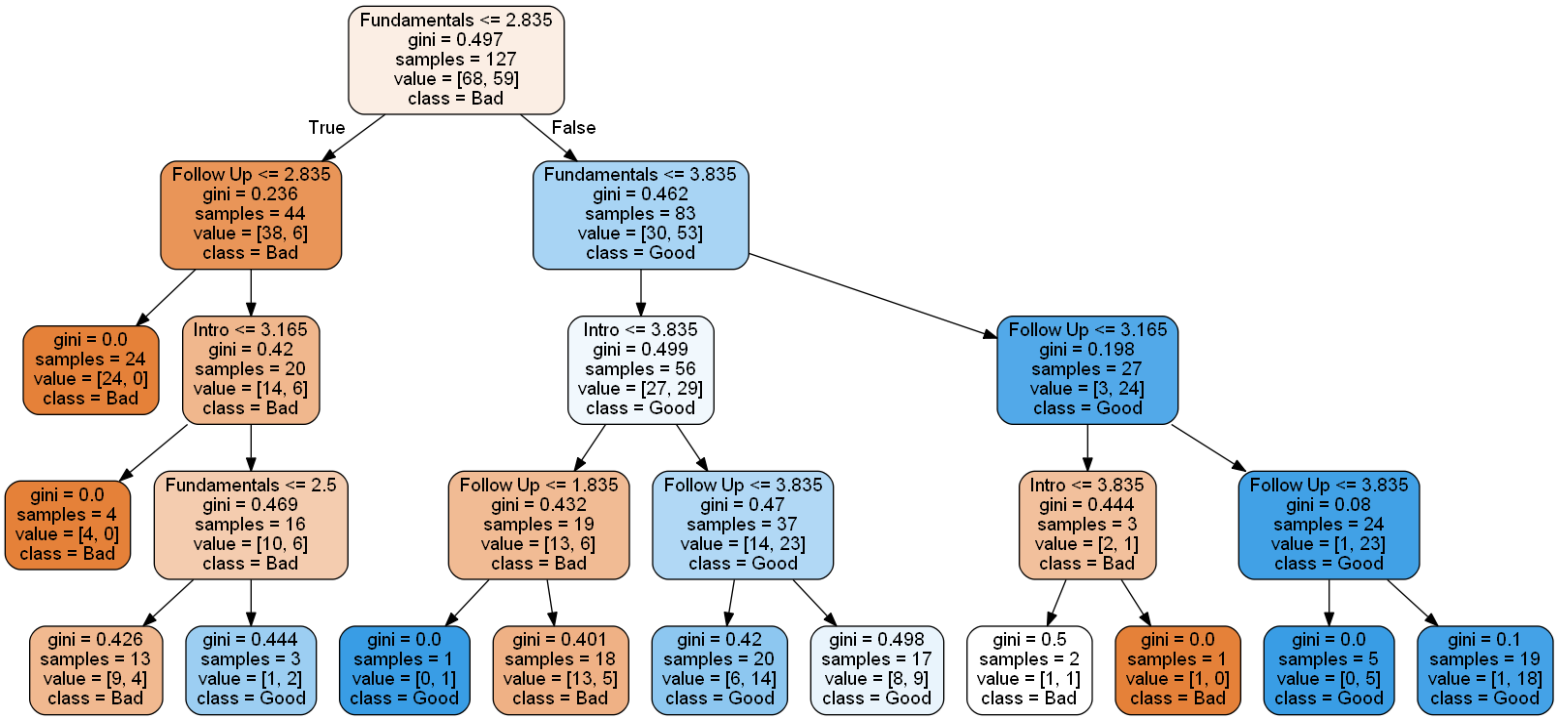
- Total Records = 196
- 65% training data = 127
- 35% testing data = 69

Feature Importance:

- Grade in Intro Course: 0.23
- Grade in Follow Up Course: 0.32
- Grade in Fundamentals Course: 0.44

Accuracy: From Random Forest Classifier

- Test Data Accuracy = 59.42%
- Train Data Accuracy = 88.18%



Analysis:

- All students who receive greater than 3.83 in Fundamentals with greater than 3.16 in Follow Up are predicted to get Good Grades
- Students who get less than or equal to 3.16 in Follow Up course with greater than 3.8 in Fundamentals course are predicted to receive Bad Grades
- Students with grades greater than 3.8 in Intro course but less than or equal to 3.8 in Fundamentals course are predicted to get Good Grades
- Students with grades less than or equal to 3.8 in Intro Course and less than or equal to 3.8 in Fundamentals Course are predicted to get Bad Grades
- Students with less than or equal to 2.83 in Fundamentals Course are predicted to receive Bad Grades regardless of grades in other courses. However, if students have received greater than 2.8 in Follow Up, greater than 3.1 in Intro with greater than 2.5 in Fundamentals, then they are predicted to get Good Grades

Hypothesis 3:

Who does better in Intro Course?

Sub data: Non-Science Students

Input Variables:

1. SAT Reading
2. SAT Math
3. High School GPA

Goal Attribute: Grade in Intro Course

- Good Grade is greater than or equal to 3.25
- Bad Grade is less than 3.25

Preprocessed Data:

- SAT Math and SAT Reading score rounded off to nearest 50 values.
  - Example: 445 becomes 450, 510 becomes 500
- High School GPA is rounded off to nearest first decimal place values.
  - Example 3.24 becomes 3.2, 2.36 becomes 2.4

Data Split:

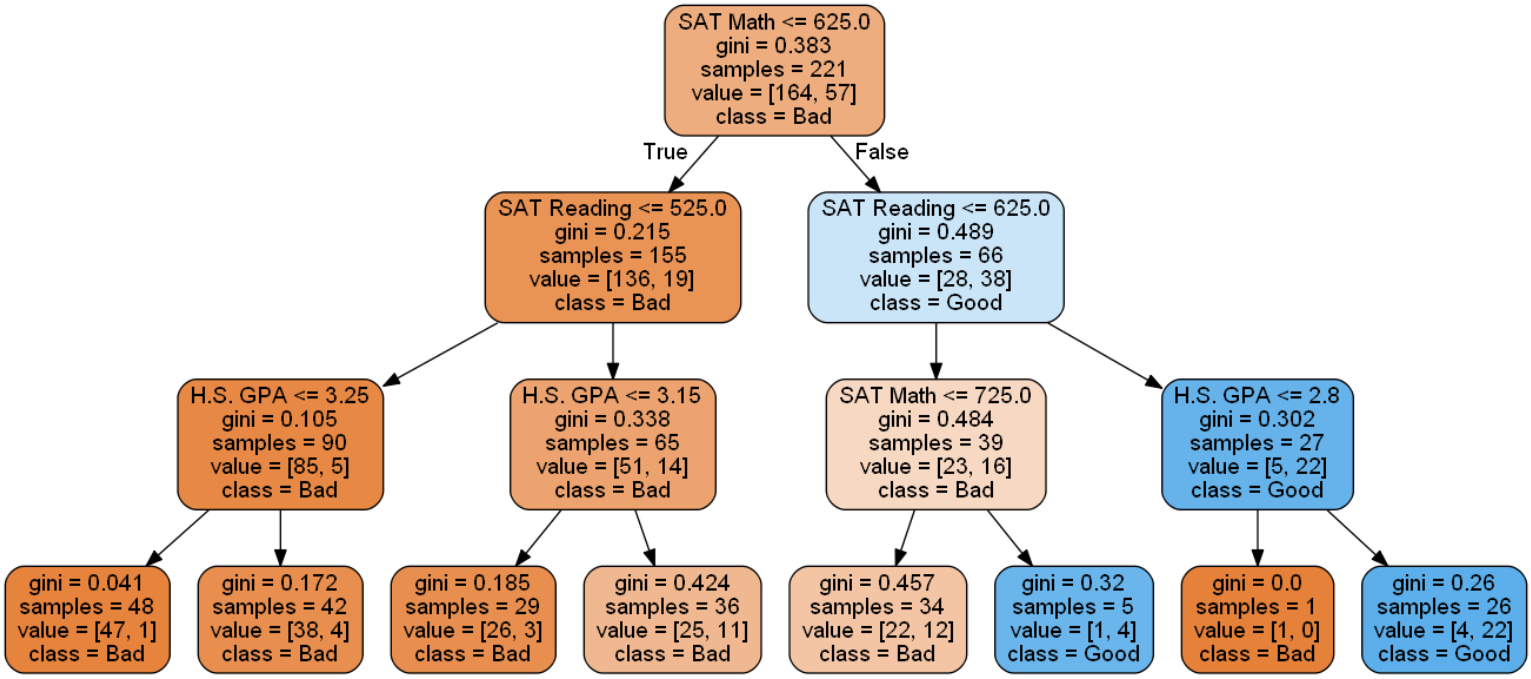
- Total records = 317
- 70% training data = 221
- 30% testing data = 96

Feature Importance:

- **SAT Reading:** 0.32
- **SAT Math:** 0.37
- **High School GPA:** 0.29

Data Accuracy:

- **Test Data Accuracy:** 79.16%
- **Train Data Accuracy:** 93.66%



Analysis:

- All students with SAT Math and SAT Reading score greater than 625 with High School GPA greater than 2.8 are predicted to get Good Grades, but the ones with same SAT scores with High School GPA less than or equal to 2.8 are predicted to get Bad Grades
- Students with less than or equal to 625 SAT Reading Score but greater than 725 SAT Math Score are predicted to get Good Grades, and the ones with grades within 625 to 725 in SAT Math score are predicted to get Bad Grades
- All Students with less than or equal to 625 SAT Math score and less than or equal to 525 SAT Reading Score are predicted to get Bad Grades
- All students with less than or equal to 625 SAT Math score but greater than 525 SAT Reading Score, regardless of their High School GPA are predicted to get Bad Grades

Hypothesis 4:

Who graduates quickly?

Sub data: Graduated students

Input Variables:

1. Sex
2. SAT Math
3. High School GPA

Goal Attribute: Semester Taken to Graduate

- Quick Graduation is less than or equal to 16 semesters
- Slow Grade is greater than 16 semesters

Preprocessed Data:

- SAT Math rounded off to nearest 50 value.
  - Example: 445 becomes 450, 510 becomes 500
- High School GPA is rounded off to nearest first decimal place.
  - Example 3.24 becomes 3.2, 2.36 becomes 2.4
- Sex set as 1 for Male, 0 for Female

Data Split:

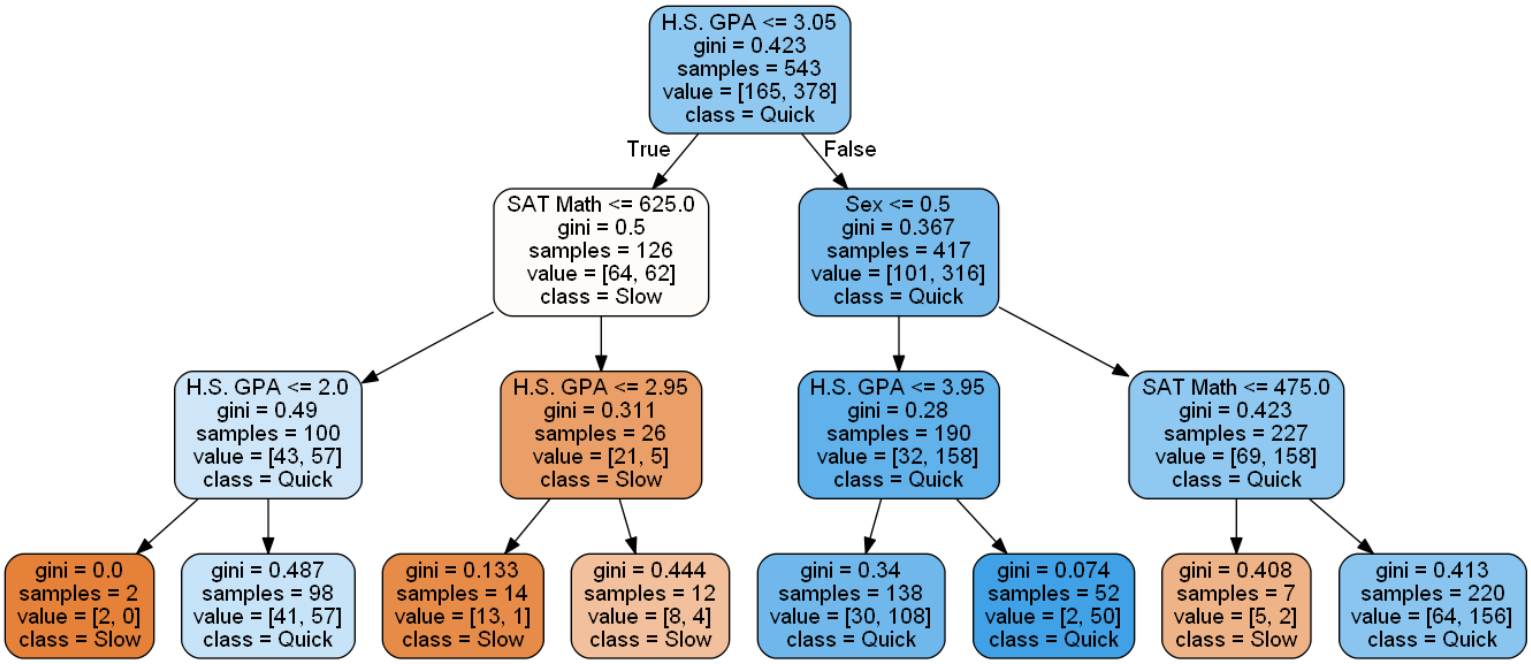
- Total records = 777
- 70% train data = 543
- 30% test data = 234

Feature Importance:

- **Sex:** 0.05
- **SAT Math:** 0.28
- **High School GPA:** 0.66

Data Accuracy:

- **Test Data Accuracy:** 63.24%
- **Train Data Accuracy:** 78.82%



Analysis:

- All Male students with High School GPA greater than 3.05 with SAT Math score greater than 475 are predicted to graduate quickly. The ones with High School GPA greater than 3.05 and SAT Math Score less than or equal to 475 are predicted to graduate slowly
- All Female students with High School GPA greater than 3.05 are predicted to graduate quickly regardless of their SAT Math Score
- All students with High School GPA less than or equal to 3.05 despite of having SAT Math score greater than 625 are predicted to graduate slowly
- All Students with SAT Math score less than or equal to 625 with High School GPA greater than 2.0 are predicted to graduate quickly. However, the ones with less than or equal to 2.0 High School GPA are predicted to graduate slowly.

Hypothesis 5:

Who does better in college?

Sub data: Race No. 5

Input Variables:

1. SAT Reading
2. SAT Math
3. High School GPA

Goal Attribute: College GPA

- Good GPA is greater than or equal to 3.25
- Bad GPA is less than 3.25

Preprocessed Data:

- SAT Math and SAT Reading rounded off to nearest 50 value.
  - Example: 445 becomes 450, 510 becomes 500
- High School GPA is rounded off to nearest first decimal place.
  - Example 3.24 becomes 3.2, 2.36 becomes 2.4

Data Split:

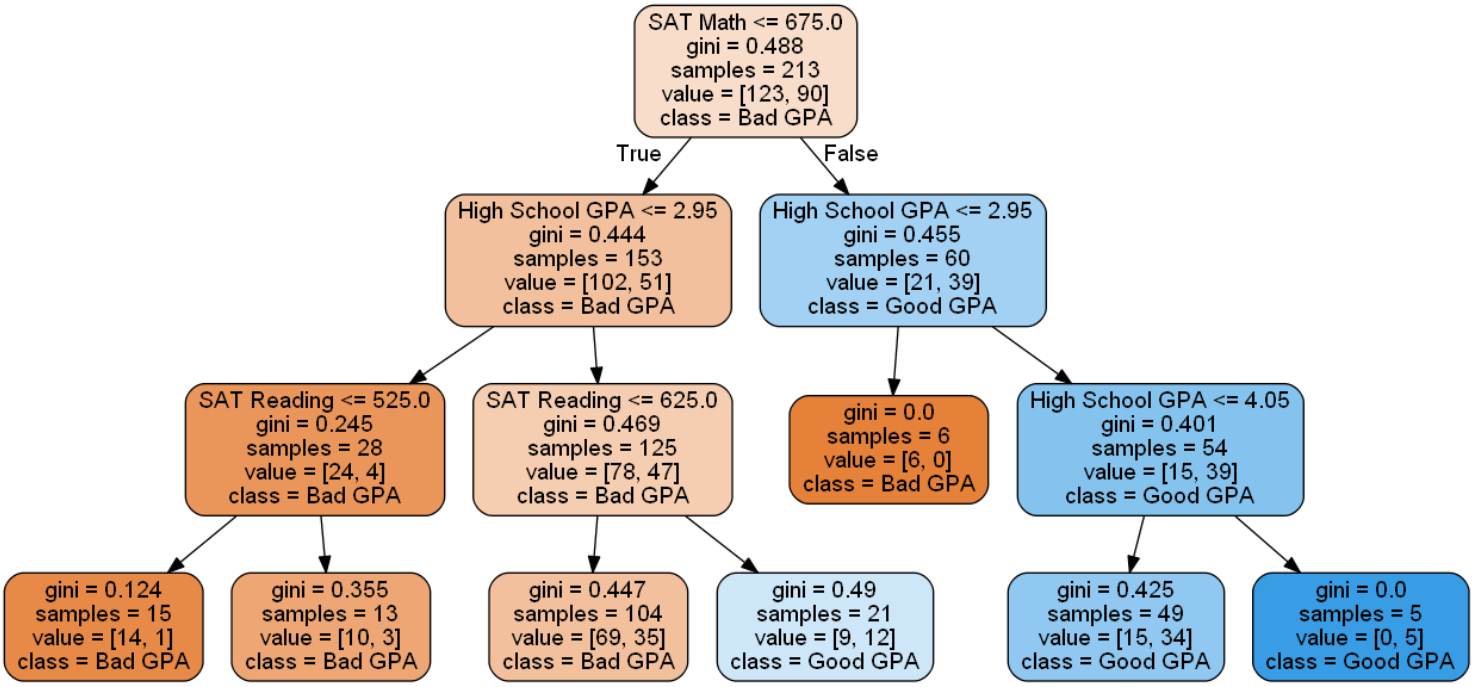
- Total records = 355
- 60% train data = 213
- 40% test data = 142

Feature Importance:

- **SAT Reading:** 0.24
- **SAT Math:** 0.23
- **High School GPA:** 0.52

Data Accuracy:

- **Test Data Accuracy:** 68.30%
- **Train Data Accuracy:** 88.73%



Analysis:

- All Students with SAT Math Score greater than 675 with High School GPA greater than 2.95 are predicted to get Good GPA.
- Students with SAT Math Score greater than 675 but High School GPA less than or equal 2.95 are predicted to get Bad GPA
- Students with SAT Math Score less than or equal to 675, but High School GPA greater than 2.95 and SAT Reading score greater than 625 are predicted to get Good GPA
- Those with SAT Reading Score less than or equal to 625 but with High School GPA greater than 2.95 with SAT Math score less than or equal to 675 are predicted to get Bad GPA
- Students with SAT Math score less than or equal to 675 and High School GPA less than or equal to 2.95are predicted to get Bad GPA regardless of their SAT Reading Score.

Source Code:

```
import pandas as pd
import numpy as np

from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier

from sklearn.model_selection import train_test_split

from pydotplus import graph_from_dot_data
from sklearn.tree import export_graphviz


def custom_round(x, base = 50):
    return int (base * round(float(x)/base))

dat = pd.read_csv('Cumulative.csv')

header = ['Id', #1 A
          'Sex', #2 B
          'Race', #3 C
          'First Gen', #4 D
          'SATReading', #5 E
          'SATMath', #6 F
          'High School GPA', #7 G
          'Semester Taken',#8 H
          'GPA',#9 I
          'Major Type',#10 J 1=CS, 2=Math, 3=Science, 4=Non-Science
          'Grade Intro Course',#11 K
          'Grade Follow Up Course',#12 L
          'Grade Fundamentals Course',#13 M
          'Grade Systems Course',#14 N
          'Grade Software Course',#15 O
          'Grade Paradigm Course',#16 P
          'Instructor Intro',#17 Q
          'Instructor Follow Up',#18 R
          'Instructor Fundamentals',#19 S
          'Instructor Systems'#20 T
         ]

dat.columns = header

#select the input variables
my_features_names = ['SATReading','SATMath','High School GPA', 'Grade Intro
Course']

#set the dataframe according to the input variables
df = pd.DataFrame(dat, columns = my_features_names)
```

```
#clear the NaN data
df = df.dropna(axis=0)

#transform the data as needed
x1 = df['SATReading'].apply((lambda x: custom_round(x, base=50))).values
x2 = df['SATMath'].apply((lambda x: custom_round(x, base=50))).values
x3 = df['High School GPA'].round(1).values

#set X data
X = pd.DataFrame(x1, columns=['SATReading'])
X['SATMath'] = x2
X['High School GPA'] = x3

#set y as target data
y = np.where(df['Grade Intro Course'] >= 3.25, 1, 0)

#split the data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.30, random_state=1, stratify=y)

print('Labels counts in (goal) attributes y:', np.bincount(y), len(y))
print('Labels counts in (goal) y_train:', np.bincount(y_train), len(y_train))
print('Labels counts in (goal) y_test:', np.bincount(y_test), len(y_test))

print('Labels counts in (features) X:', len(X))
print('Labels counts in (features) X_train:', len(X_train))
print('Labels counts in (features) X_test:', len(X_test))


#*****************************************************************************
forest = RandomForestClassifier(criterion='gini',
                                n_estimators=25,
                                random_state=1,
                                n_jobs=2)
forest.fit(X_train, y_train)
print("Importances: ", list(zip(X_train, forest.feature_importances_)))

#see accuracy
from sklearn.metrics import accuracy_score

#predict out of the training data set
test_preds = forest.predict(X_test)
train_preds = forest.predict(X_train)
#get the actual goal attributes of the testing data set
test_true = y_test
train_true = y_train
print("Test Data Accuracy: ", accuracy_score(test_true, test_preds))
print("Train Data Accuracy: ", accuracy_score(train_true, train_preds))
```

```
#*****************************************************************************
tree = DecisionTreeClassifier(criterion='gini',
                              max_depth=3,
                              random_state=1)
tree.fit(X_train, y_train)

X_combined = np.vstack((X_train, X_test))
y_combined = np.hstack((y_train, y_test))


dot_data = export_graphviz(tree,
                           filled=True,
                           rounded=True,
                           class_names=['Bad',
                                        'Good'],
                           feature_names=['SAT Reading',
                                        'SAT Math',
                                        'H.S. GPA'],
                           out_file=None)
graph = graph_from_dot_data(dot_data)
graph.write_png('3.png')
```