The University of Oklahoma

Final Project Report

High School Student Performance Analysis

Team 2

Data Science and Analytics - MIT-5742-930

Dr. Heshan Sun

December 4th, 2024

High School Student Performance Analyses

1. The Problem Addressed

This study examines the factors that impact high school students' academic performance, focusing on their Grade Point Average (GPA). Key elements such as tutoring, weekly study time, and school absences are analyzed to understand their influence on GPA. The research aims to identify actionable insights that educators can use to improve student outcomes. By pinpointing these factors, the study provides a foundation for targeted support and intervention strategies. Insights from this analysis can help schools design programs that foster academic success. Ultimately, the findings aim to guide policymakers in implementing measures to boost overall student performance.

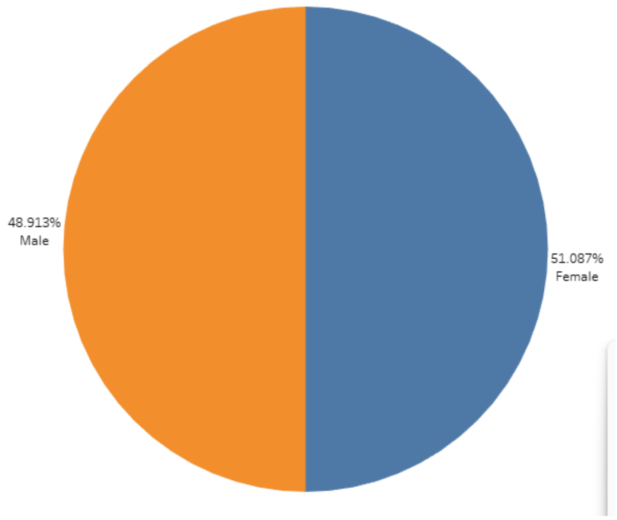
2. Description of the Data

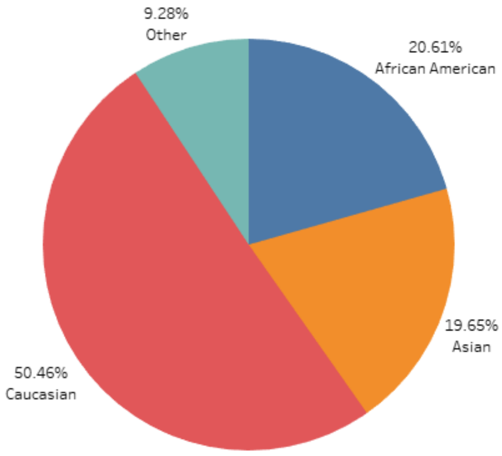
The dataset comprises 2,392 student records and includes attributes like:

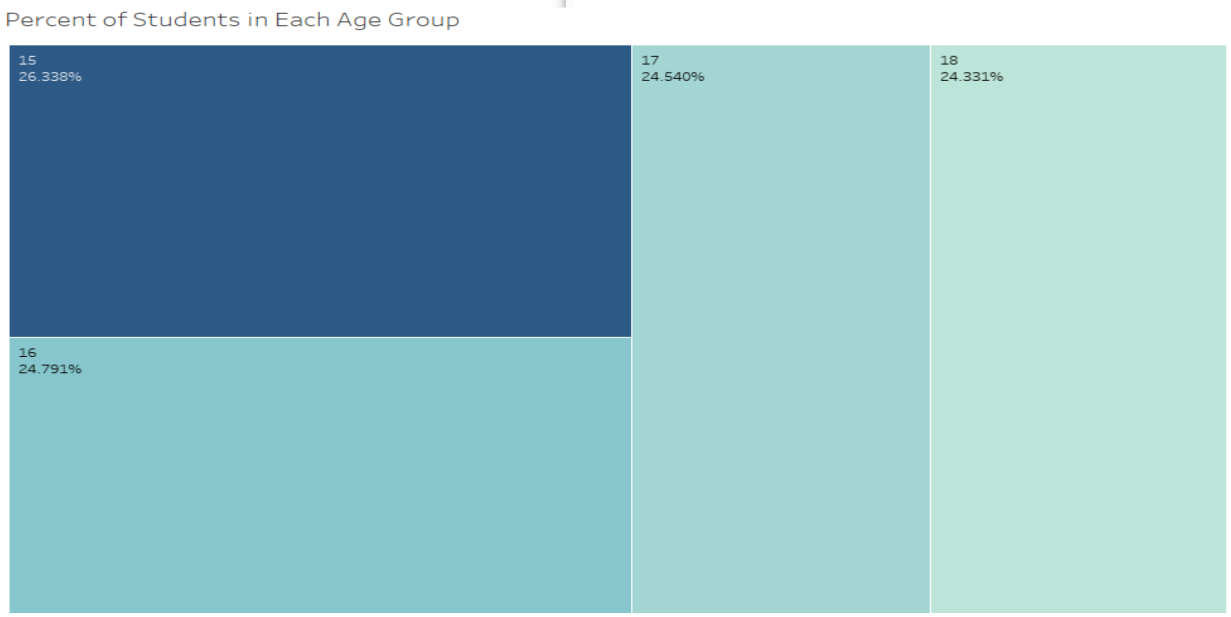
- Tutoring: Indicates if a student receives additional support.
- StudyTimeWeekly: Average weekly study hours.
- Absences: Number of school days missed.
- GPA: Target variable representing academic performance.

Descriptive Statistics and Visualizations and Insights from Initial Analysis

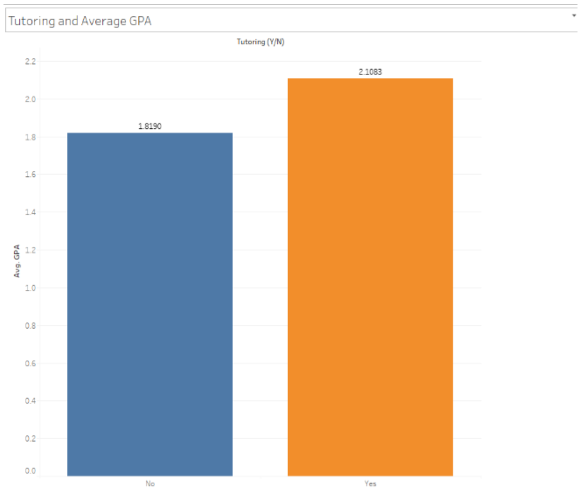
The dataset includes 2,392 student records, each representing a unique student, with a mix of categorical and numerical attributes to explore the effects of study habits and support systems on academic performance. The student population is almost evenly split between genders. Regarding race demographics, 50% are Caucasian, 20.61% African American, 19.66% Asian, and 9.28% from other racial groups. Students' ages range from 15 to 18 years, distributed nearly equally across this range. Descriptive analysis, including pie and bar charts, was conducted to better understand the dataset and uncover insights for the model analysis. See Appendix A & B for further information.

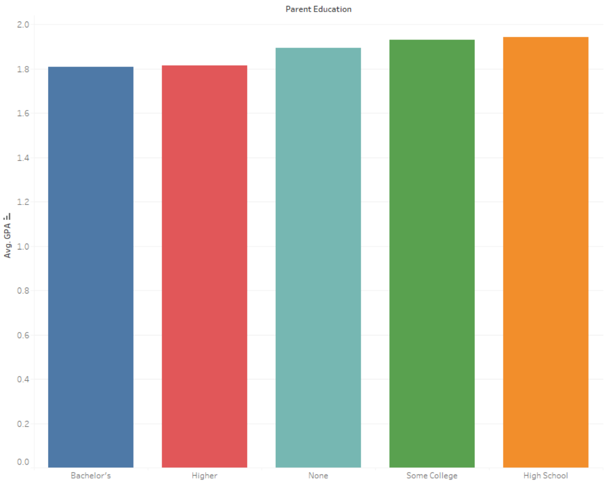






In the graphs above we can see the data set is almost split evenly for males and females, with slightly more females. Over 50% of the students are Caucasian, 20.61% African American, 19.66 Asian, and 9.28% other races. The students' age ranges from 15-18 with almost equal number of students in each age group.





In the graphs above we analyze how Tutoring and Parental Education affect the students' GPA. The graphs show that students with tutoring had a 0.30 higher GPA than students with no tutoring. In addition, we see that students with parents that received high school education have the highest GPA on average.

3. Procedure Followed for Analyses

The analysis was conducted in four key steps to develop predictive models and insights for student GPA:

Step 1: Data Preprocessing

The dataset was cleaned by removing irrelevant features, and relevant variables were selected. For the scikit-learn analysis, the chosen predictors were Tutoring, StudyTimeWeekly, and Absences, with GPA as the target variable. The Excel regression analysis focused on the relationship between Absences and GPA.

Step 2: Data Splitting

The dataset was divided into training (67%) and test (33%) subsets. This split ensured sufficient data for model training while reserving enough unseen data for evaluation.

Step 3: Model Training

Linear regression models were developed using both scikit-learn and Excel:

- **Scikit-learn Linear Regression** analyzed the combined impact of Tutoring, StudyTimeWeekly, and Absences on GPA.
- **Excel Regression** specifically explored the relationship between Absences and GPA.

Step 4: Evaluation Metrics

The models were evaluated using the following metrics:

- R-squared (R^2): To measure the proportion of variance in GPA explained by the model.
- Mean Squared Error (MSE): To assess prediction accuracy.
- OLS Analysis: P-values for each coefficient were calculated using statsmodels to determine the statistical significance of the predictors.

These steps ensured a thorough analysis of the factors affecting student GPA, highlighting their predictive power and statistical relevance.

4. **Major Results**

The analysis identified significant factors influencing student GPA through linear and logistic regression models, providing a comprehensive understanding of academic performance predictors.

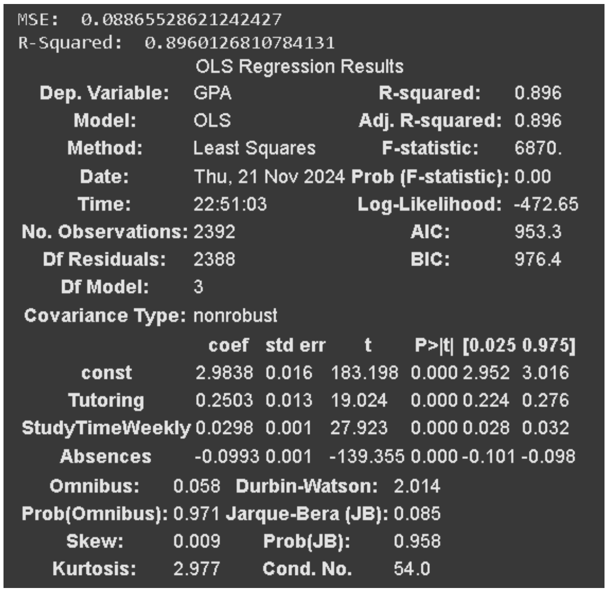
Scikit-learn Linear Regression Findings

The scikit-learn model achieved an R-squared value of 0.896, indicating that approximately 89.6% of the variance in GPA is explained by the predictors. Key coefficients and their interpretations are as follows:

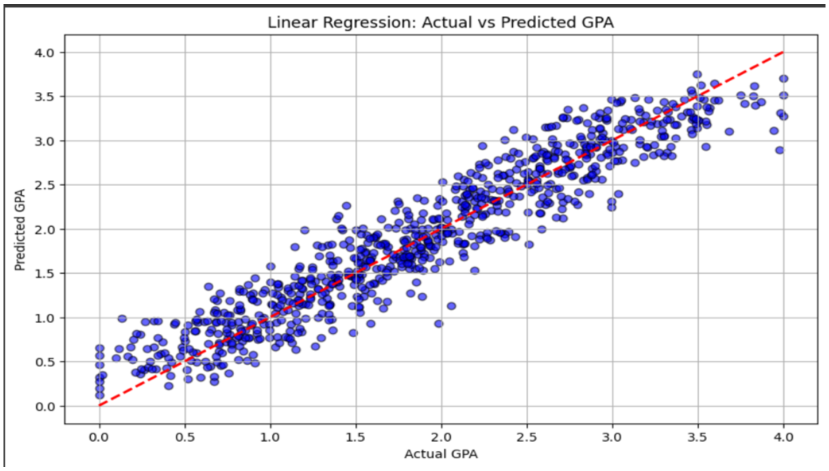
- Tutoring: Coefficient +0.2503, p-value 0.000. Students receiving tutoring experienced an average GPA increase of 0.25 points, a statistically significant effect. This finding aligns with studies reporting a strong positive impact of tutoring on academic performance.

- StudyTimeWeekly: Coefficient +0.0298, p-value 0.000. Weekly study hours showed a smaller but significant positive impact on GPA.
- Absences: Coefficient -0.0993, p-value 0.000. Each additional absence corresponded to a significant GPA decrease of 0.10 points.

- **Model Performance:** The R² score was 0.896, indicating that 89.6% of the variance in GPA could be explained by the model. The Mean Squared Error (MSE) was 0.0887, which is relatively low, indicating that the model predictions are close to the actual values



```
MSE:  0.08865528621242427
R-Squared:  0.8960126810784131
                    OLS Regression Results
   Dep. Variable:   GPA                     R-squared:     0.896
        Model:      OLS                 Adj. R-squared:    0.896
       Method:      Least Squares           F-statistic:   6870.
        Date:       Thu, 21 Nov 2024 Prob (F-statistic):   0.00
        Time:       22:51:03              Log-Likelihood:  -472.65
No. Observations:   2392                         AIC:      953.3
   Df Residuals:    2388                         BIC:      976.4
     Df Model:      3
Covariance Type:    nonrobust
                     coef   std err     t      P>|t|  [0.025  0.975]
      const         2.9838  0.016    183.198   0.000  2.952   3.016
     Tutoring       0.2503  0.013    19.024    0.000  0.224   0.276
StudyTimeWeekly     0.0298  0.001    27.923    0.000  0.028   0.032
     Absences      -0.0993  0.001   -139.355   0.000 -0.101  -0.098
    Omnibus:     0.058  Durbin-Watson:   2.014
Prob(Omnibus):   0.971  Jarque-Bera (JB): 0.085
      Skew:      0.009     Prob(JB):     0.958
    Kurtosis:    2.977     Cond. No.     54.0
```

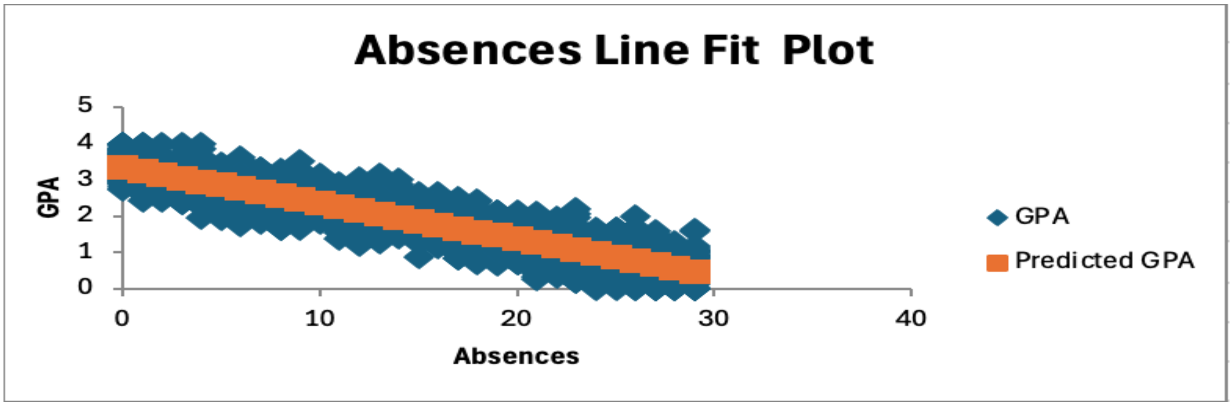


Excel Linear Regression Findings

The Microsoft Excel linear regression model focused on the relationship between Absences and GPA, yielding the following results:

- R-Squared: 0.845 – This indicates that approximately 84.5% of the variance in GPA is explained by student absences.
- Coefficient for Absences: -2.775, with a p-value of 0.000. This strong negative coefficient indicates that increased absences significantly decrease GPA.

These findings align with existing research, such as the study by Aucejo & Romano (2016), which concluded that students with higher absenteeism miss valuable learning opportunities, leading to poor academic performance. The analysis further supports the hypothesis that students with fewer absences are more likely to perform better academically and achieve a higher GPA. Below is the result of the linear regression model of GPA and absences conducted through Excel:



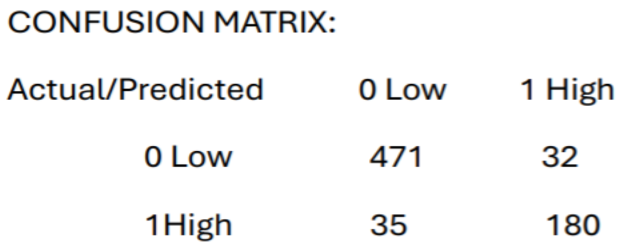
Logistic Regression Findings

The logistic regression model provided additional insights into categorical predictors of GPA, with the following key findings.

Evaluation Metrics:

- Accuracy: 90.67% – The model correctly predicted 471 true negatives and 180 true positives.
- Sensitivity: 83.7% – The model identified 83.7% of high-performing students.
- Specificity: 93.6% – The model accurately identified 93.6% of low-performing students.
- Akaike Information Criterion (AIC): 1006.5, indicating a well-fitting model.

Confusion Matrix:



CONFUSION MATRIX:

Actual/Predicted	0 Low	1 High
0 Low	471	32
1High	35	180

Key Predictors:

- Tutoring: Coefficient +0.813, significant positive impact on GPA.
- StudyTimeWeekly: Coefficient +0.507, moderate positive effect on GPA.
- Absences: Coefficient -2.775, strong negative impact on GPA.
- Parental Support: Coefficient +0.403, positive contribution to GPA.
- Extracurriculars: Coefficient +0.607, meaningful positive effect on GPA.

Insignificant Predictors: Age, Gender, Ethnicity, ParentalEducation, Music, Volunteering.

These findings emphasize the importance of Tutoring, Study Time, Attendance, Parental Support, and Extracurricular Activities in predicting student GPA. The logistic regression model adds a valuable dimension to understanding the factors that influence academic performance, providing actionable insights for educators and policymakers.

5. **Conclusions & Key Findings**

The analysis revealed several key factors influencing student GPA. Tutoring and weekly study time were found to have a positive impact on GPA, with students who received tutoring and devoted more time to studying each week generally performing better academically. Conversely, absences showed a strong negative correlation with GPA. Students with higher absenteeism experienced a significant decline in their academic performance, highlighting the crucial role of consistent attendance. The R^2 values from both the linear and logistic regression models were high, indicating that the models provided a good fit to the data and could effectively predict GPA based on these key factors.

Recommendations for Stakeholders

To improve student academic performance, institutions should prioritize expanding access to tutoring programs, as it has a clear positive effect on GPA. Additionally, fostering consistent study habits should be emphasized, encouraging students to dedicate regular time to their studies in order to achieve better academic results. Reducing absenteeism should also be a focus, as it is strongly correlated with lower GPA. Schools can implement targeted interventions, such as offering attendance incentives or providing additional support to students who face challenges in maintaining regular attendance, in order to mitigate the negative impact of absences on their academic success.

Implications for future work

The models developed in this study effectively identify the key factors that contribute to student academic performance. Moving forward, future analyses could benefit from expanding the dataset to include additional factors, such as socioeconomic status, parental involvement, and other support systems, which might provide even deeper insights into the complexities of student performance. These expanded models could help institutions design more targeted and comprehensive strategies to support student success.

References

Aucejo, E. M., & Romano, T. F. (2016). Assessing the effect of school days and absences on test score performance. Economics of Education Review, 55, 70-87. https://doi.org/10.1016/j.econedurev.2016.08.007

Nickow, A., Oreopoulos, P., & Quan, V. (2020). The impressive effects of tutoring on prek-12 learning: A systematic review and meta-analysis of the experimental evidence. NBER. https://www.nber.org/papers/w27476

Rabie El Kharoua. (2024). 📚 Students Performance Dataset 📚 [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DS/5195702

Appendix A - Columns – High School Student Performance

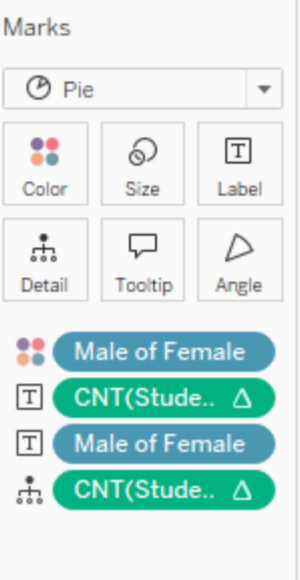
Name	Type	Usage Notes
student_id	Number	Not Used – unique
age	Number	Used
gender	Number	Used
ethnicity	Number	Used
parental_education	Number	Used
study_time_weekly	Number	Used
absences	Number	Used
tutoring	Number	Used
parental_support	Number	Used
extracurricular	Number	Used
sports	Number	Used
music	Number	Used
volunteering	Number	Used
gpa	Number	Used
grade_class	Number	Used

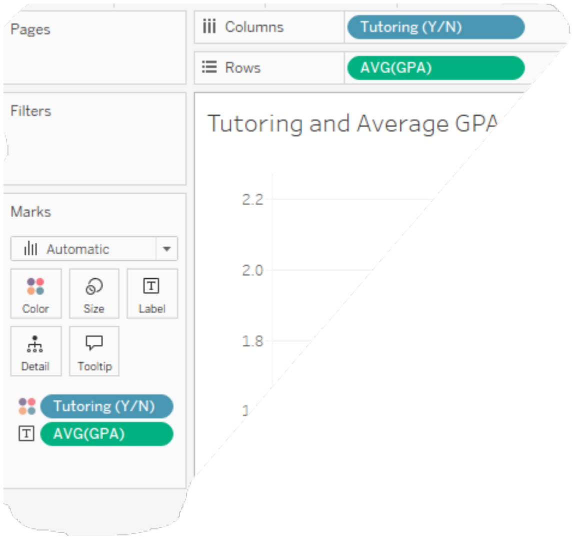
Appendix B – Simplified Overview of High School Student Performance Data

- **Ethnicity** – 0: Caucasian, 1: African American, 2: Asian, 3: Other
- **Parental Education** – 0: None, 1: High School, 2: Some College, 3: Bachelor's, 4: Higher
- **Parental Support** – 0: None, 1: Low, 2: Moderate, 3: High, 4: Very High
- **Grade Class** – 0: 'A' (GPA >= 3.5), 1: 'B'(3.0 <= GPA < 3.5), 2: 'C' (2.5 <= GPA < 3.0), 3: 'D' (2.0 <= GPA < 2.5), 4: 'F'(GPA < 2.0)
- **Other columns** – 0: No, 1: Yes

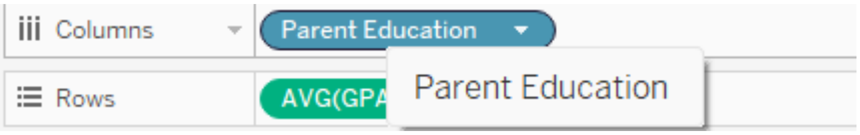
Appendix C – Methods of Analyses

Tableau Charts - Female and Male Pie Chart - Tutoring and Average GPA

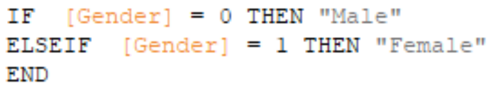




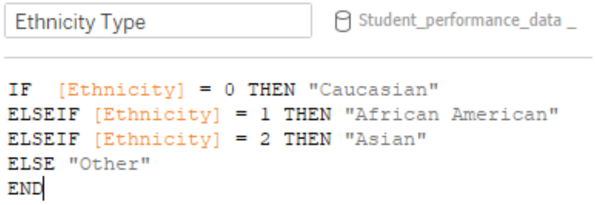
GPA by Parental Education   % of Students in Each Age Group (Tree Map) & Student



Ethnicity Type Pie Chart

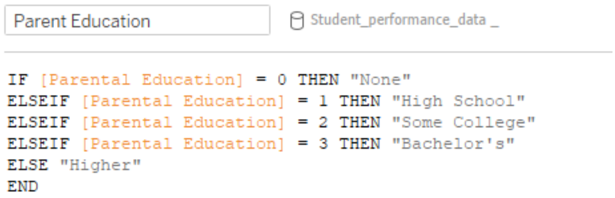


```
IF  [Gender] = 0 THEN "Male"
ELSEIF  [Gender] = 1 THEN "Female"
END
```



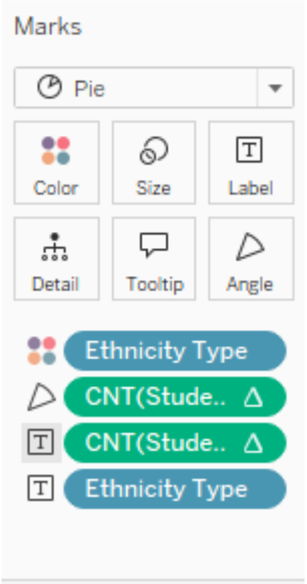
Ethnicity Type   Student_performance_data _

```
IF  [Ethnicity] = 0 THEN "Caucasian"
ELSEIF [Ethnicity] = 1 THEN "African American"
ELSEIF [Ethnicity] = 2 THEN "Asian"
ELSE "Other"
END
```



Parent Education   Student_performance_data _

```
IF [Parental Education] = 0 THEN "None"
ELSEIF [Parental Education] = 1 THEN "High School"
ELSEIF [Parental Education] = 2 THEN "Some College"
ELSEIF [Parental Education] = 3 THEN "Bachelor's"
ELSE "Higher"
END
```

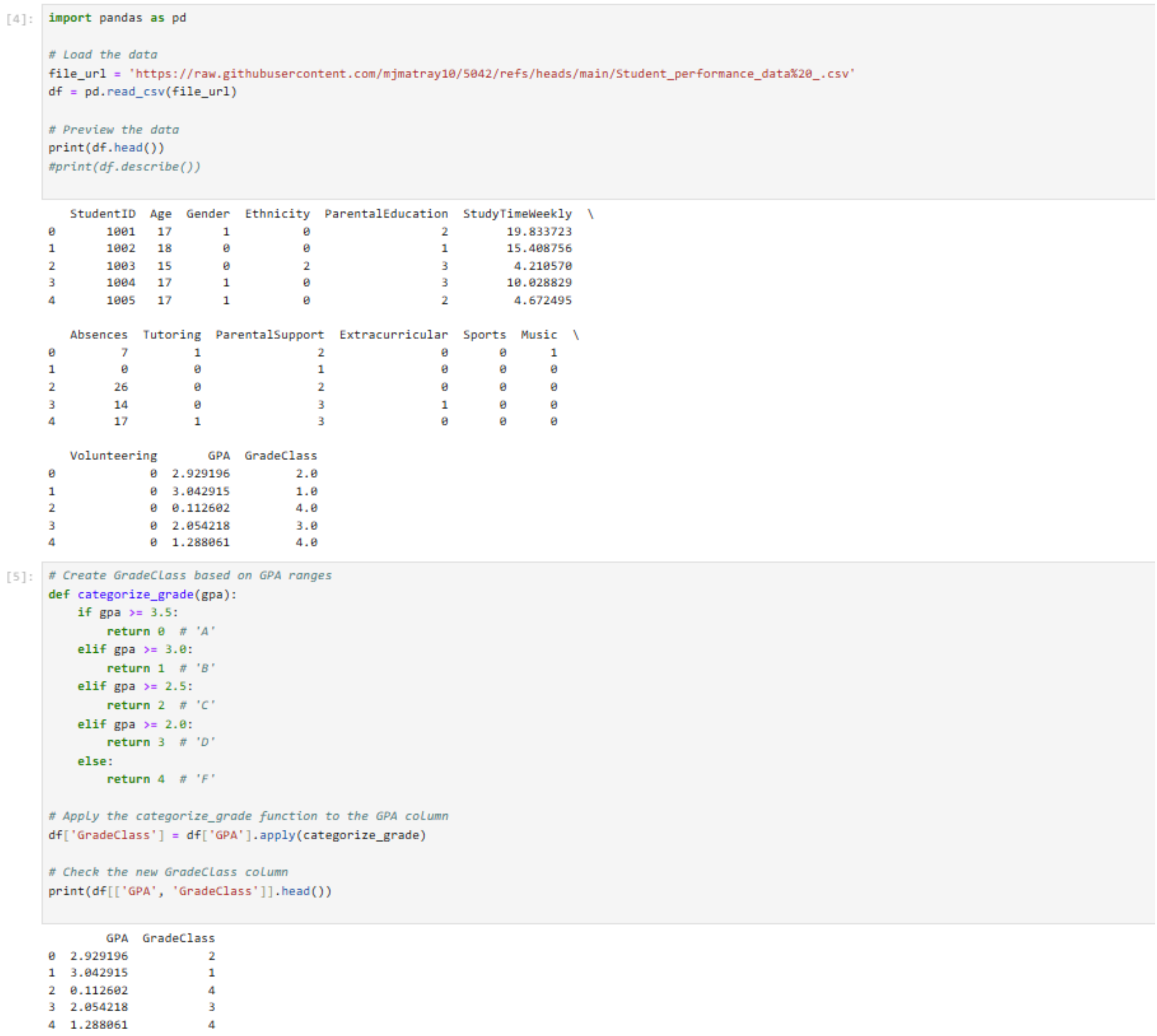


Scikit-learn & Excel Linear Regression

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score


#Generating data
import pandas as pd


# Hypothesis: "students with tutoring, more study time, and fewer absences have a better GPA"

# Replace with your file URL
file_url = 'https://raw.githubusercontent.com/mjmatray10/5042/refs/heads/main/Student_performance_data%20_.csv'
df = pd.read_csv(file_url)
df

type(df)

df.head()

df.tail()

df.describe()

features = ['Tutoring', 'StudyTimeWeekly', 'Absences']
X = df[features]
X

target = 'GPA'
y = df[target]
y

# Use sklearn package for training the lienar regression model.

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=100)

# Train linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Prediction on the test set
```

```
y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)  # comparing the model predictions and actual values
r2 = r2_score(y_test, y_pred)

print("MSE: ", mse)
print("R-Squared: ", r2)

import statsmodels.api as sm

# Add a constant to the features for the intercept
X_with_const = sm.add_constant(X)

# Fit the OLS model
ols_model = sm.OLS(y, X_with_const).fit()

# Get the summary of the OLS model, which includes p-values
ols_summary = ols_model.summary()

# Display the summary
ols_summary

import matplotlib.pyplot as plt
import numpy as np

# Create a scatter plot of actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', edgecolor='k', alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='red', linestyle='--',
linewidth=2)
plt.xlabel("Actual GPA")
plt.ylabel("Predicted GPA")
plt.title("Linear Regression: Actual vs Predicted GPA")
plt.grid(True)
plt.show()
```

Decision Tree Classifier Code



[4]:

```
import pandas as pd

# Load the data
file_url = 'https://raw.githubusercontent.com/mjmatray10/5042/refs/heads/main/Student_performance_data%20_.csv'
df = pd.read_csv(file_url)

# Preview the data
print(df.head())
#print(df.describe())
```

```
   StudentID  Age  Gender  Ethnicity  ParentalEducation  StudyTimeWeekly  \
0       1001   17       1          0                  2        19.833723   
1       1002   18       0          0                  1        15.408756   
2       1003   15       0          2                  3         4.210570   
3       1004   17       1          0                  3        10.028829   
4       1005   17       1          0                  2         4.672495   

   Absences  Tutoring  ParentalSupport  Extracurricular  Sports  Music  \
0         7         1                2                0       0      1   
1         0         0                1                0       0      0   
2        26         0                2                0       0      0   
3        14         0                3                1       0      0   
4        17         1                3                0       0      0   

   Volunteering       GPA  GradeClass  
0             0  2.929196         2.0  
1             0  3.042915         1.0  
2             0  0.112602         4.0  
3             0  2.054218         3.0  
4             0  1.288061         4.0  
```

[5]:

```
# Create GradeClass based on GPA ranges
def categorize_grade(gpa):
    if gpa >= 3.5:
        return 0  # 'A'
    elif gpa >= 3.0:
        return 1  # 'B'
    elif gpa >= 2.5:
        return 2  # 'C'
    elif gpa >= 2.0:
        return 3  # 'D'
    else:
        return 4  # 'F'

# Apply the categorize_grade function to the GPA column
df['GradeClass'] = df['GPA'].apply(categorize_grade)

# Check the new GradeClass column
print(df[['GPA', 'GradeClass']].head())
```

```
        GPA  GradeClass
0  2.929196           2
1  3.042915           1
2  0.112602           4
3  2.054218           3
4  1.288061           4
```

[17]:

```python
from sklearn.preprocessing import LabelEncoder

# Select features (excluding StudentID and GradeClass)
features = df.loc[:, ['Age', 'Gender', 'Ethnicity', 'ParentalEducation',
                      'StudyTimeWeekly', 'Absences', 'Tutoring',
                      'ParentalSupport', 'Extracurricular', 'Sports',
                      'Music', 'Volunteering']]

# Handle categorical features (encode categorical variables)
label_encoders = {}
for column in ['Gender', 'Ethnicity', 'ParentalEducation']:
    le = LabelEncoder()
    features[column] = le.fit_transform(features[column])
    label_encoders[column] = le

# Target is the GradeClass
target = df['GradeClass']
```

[18]:

```python
from sklearn.model_selection import train_test_split

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.33, random_state=100)

# Preview the split data shapes
print(f"Training Data Shape: {X_train.shape}, Testing Data Shape: {X_test.shape}")
```

```
Training Data Shape: (1602, 12), Testing Data Shape: (790, 12)
```

[19]:

```python
from sklearn.tree import DecisionTreeClassifier

# Train the Decision Tree Classifier
classifier_dt = DecisionTreeClassifier(criterion='entropy', random_state=100, max_depth=5)
clf = classifier_dt.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)
```

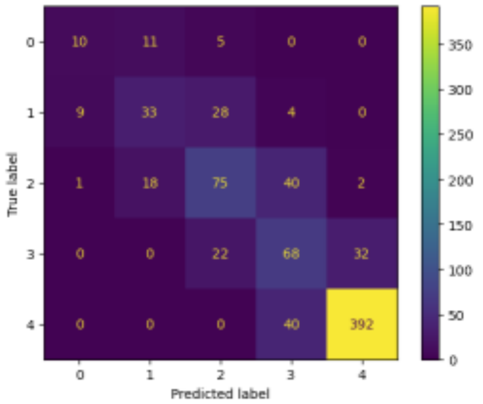
[20]:

```python
from sklearn.metrics import accuracy_score, confusion_matrix, ConfusionMatrixDisplay

# Model performance (accuracy)
performance_of_model = accuracy_score(y_test, y_pred)
print(f'Accuracy: {performance_of_model:.2f}')

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_)
disp.plot()
```

```
Accuracy: 0.73
```

[20]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2361c39c5d0>
```



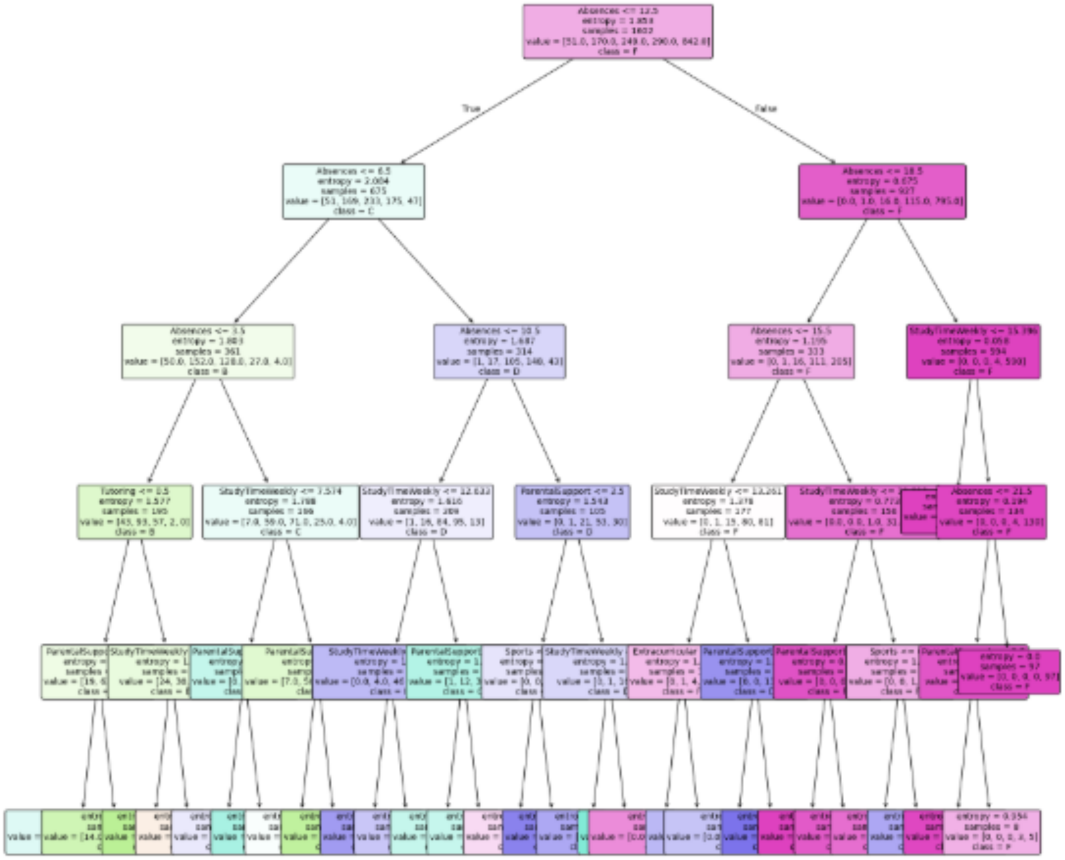
[21]:

```python
# Example prediction (input for a new student)
example_input = [[17, 1, 0, 2, 15, 5, 1, 3, 0, 0, 1, 0]]  # Input: [Age, Gender, Ethnicity, ParentalEducation, StudyTimeWeekly,
prediction = clf.predict(example_input)                  #--> Absences, Tutoring, ParentalSupport, Extracurricular, Sports, Music, Volunteering]

print(f"Predicted GPA Category: {prediction[0]}")  # Output the predicted category
```

```
Predicted GPA Category: 1
```

•[22]:

```
import matplotlib.pyplot as plt
from sklearn import tree
# Define the class labels corresponding to the GradeClass categories
class_labels = ['A', 'B', 'C', 'D', 'F']

# Visualize the decision tree
_, ax = plt.subplots(figsize=(20, 20))  # Adjust the figure size for clarity
tree.plot_tree(
    clf,
    filled=True,               # Fill the nodes with color
    feature_names=features.columns,  # Use feature names for better understanding
    class_names=class_labels,   # Display the target class names (GradeClass categories)
    rounded=True,              # Rounded corners for the nodes
    fontsize=10                # Adjust font size to fit the tree
)
plt.show()  # Display the tree visualization
```



•[23]:

```
# Get the importance of each feature
feature_importances = clf.feature_importances_
# Display the importance of each feature
for feature, importance in zip(features.columns, feature_importances):
    print(f'{feature}: {importance:.4f}')
```

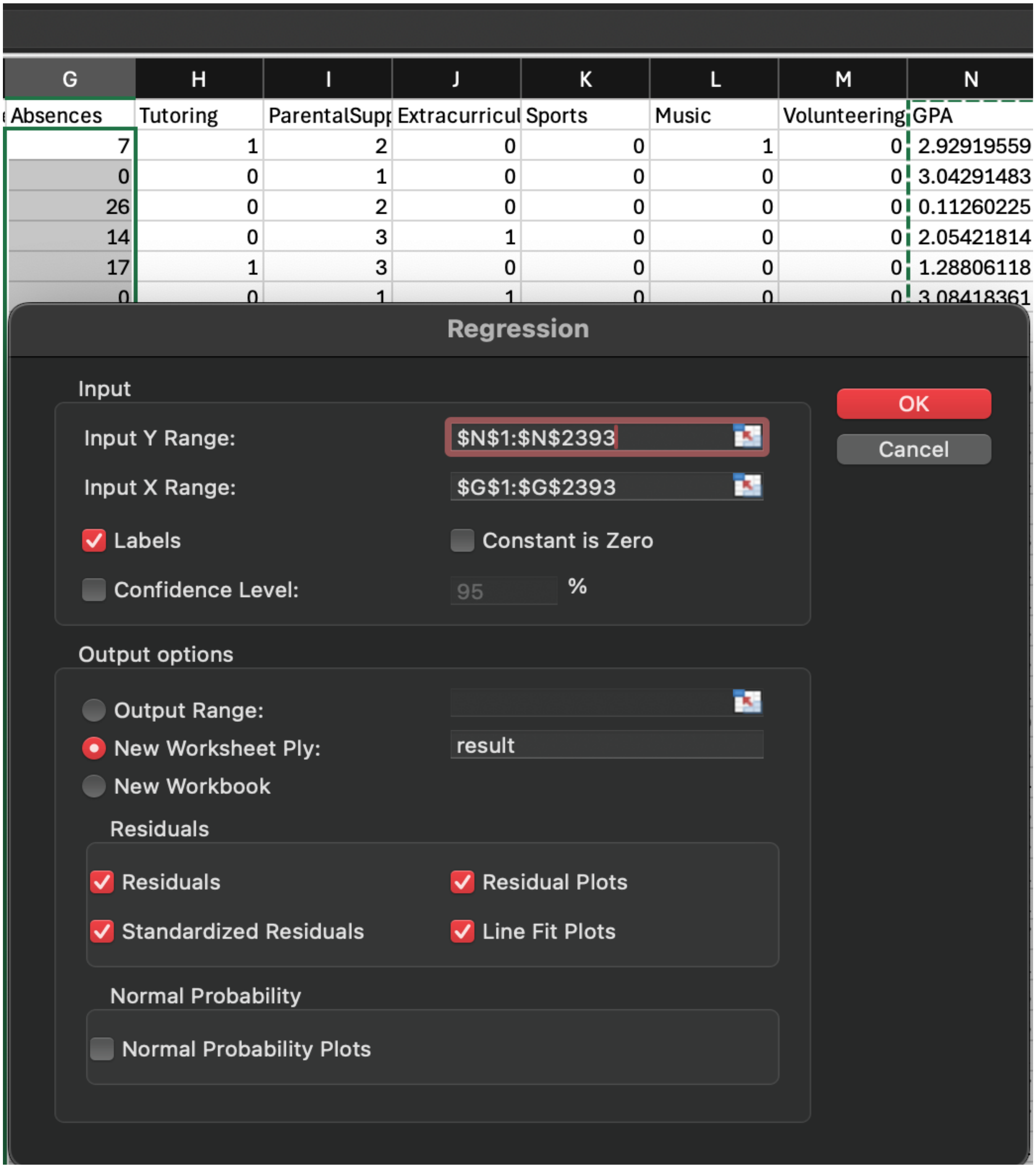
```
Age: 0.0000
Gender: 0.0000
Ethnicity: 0.0000
ParentalEducation: 0.0000
StudyTimeWeekly: 0.0816
Absences: 0.8367
Tutoring: 0.0120
ParentalSupport: 0.0536
Extracurricular: 0.0063
Sports: 0.0098
Music: 0.0000
Volunteering: 0.0000
```

[24]:

```
# Get the depth of the tree to make sure the model doesn't face overfitting
depth = clf.get_depth()
print(f'Depth of the tree: {depth}')
```

```
Depth of the tree: 5
```

Microsoft Excel Linear Regression – GPA & Absences



G	H	I	J	K	L	M	N
Absences	Tutoring	ParentalSupp	Extracurricul	Sports	Music	Volunteering	GPA
7	1	2	0	0	1	0	2.92919559
0	0	1	0	0	0	0	3.04291483
26	0	2	0	0	0	0	0.11260225
14	0	3	1	0	0	0	2.05421814
17	1	3	0	0	0	0	1.28806118
0	0	1	1	0	0	0	3.08418361

Regression

Input

Input Y Range: $N$1:$N$2393

Input X Range: $G$1:$G$2393

☑ Labels ☐ Constant is Zero

☐ Confidence Level: 95 %

Output options

○ Output Range:

● New Worksheet Ply: result

○ New Workbook

Residuals

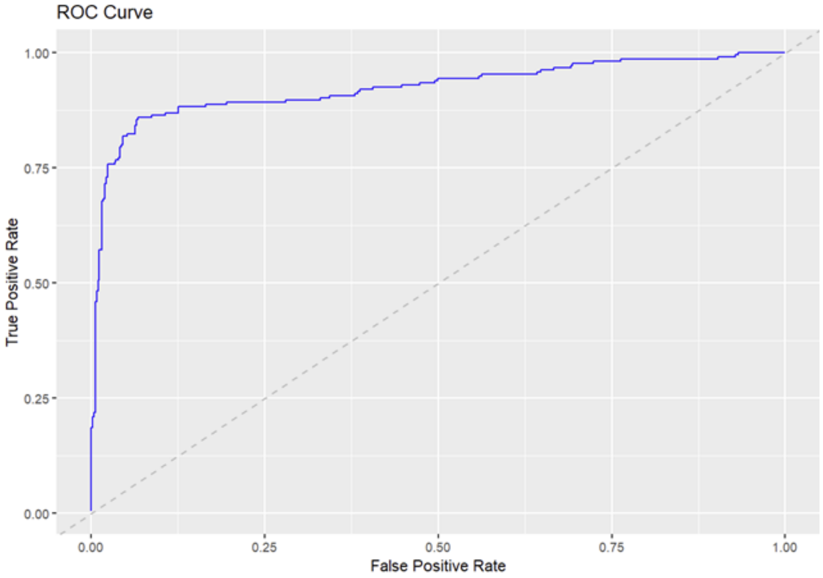
☑ Residuals ☑ Residual Plots

☑ Standardized Residuals ☑ Line Fit Plots

Normal Probability

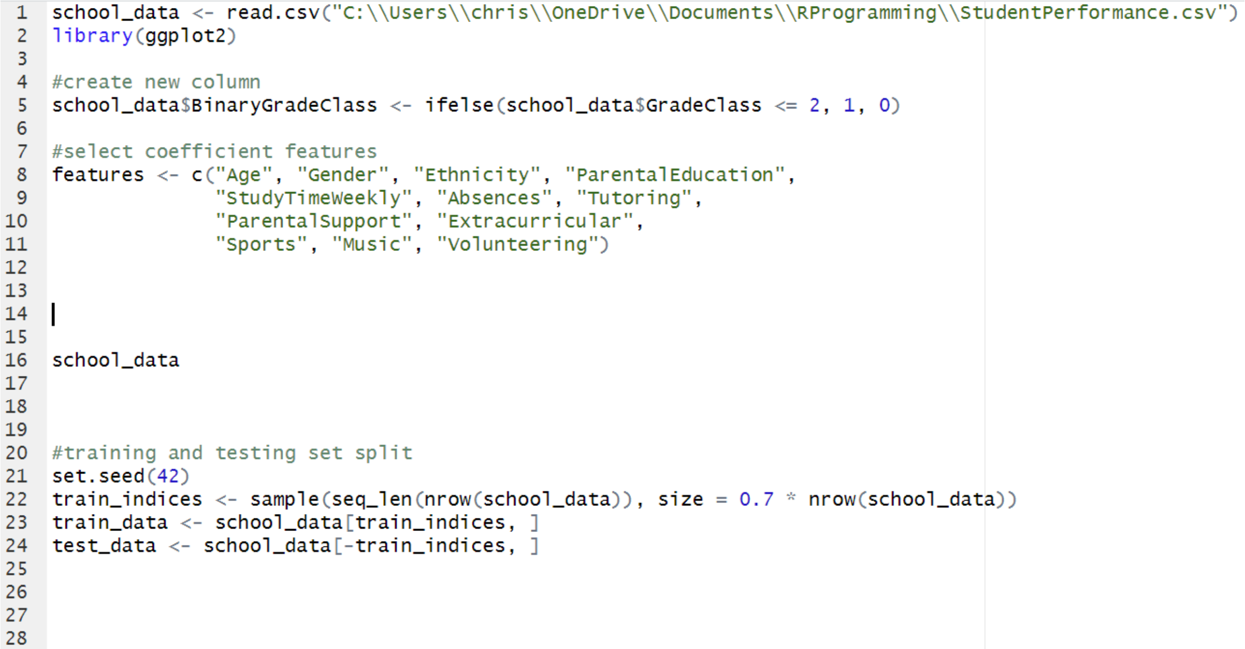
☐ Normal Probability Plots

Logistic Regression

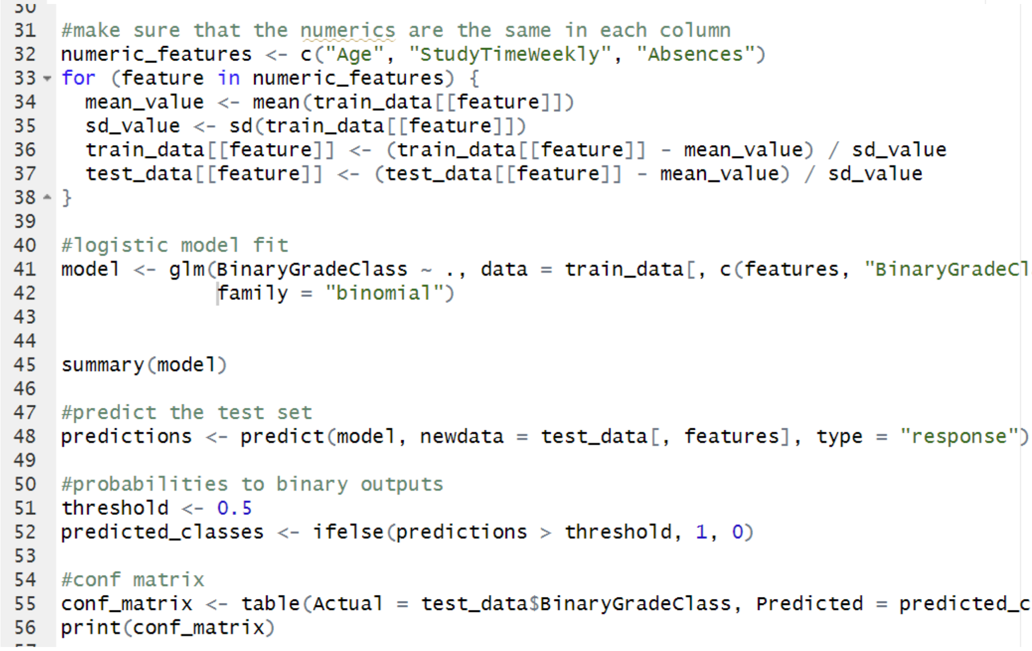


$$\log(p/1-p)=-3.087-0.028\cdot\text{Age}-0.049\cdot\text{Gender}+0.039\cdot\text{Ethnicity}-0.063\cdot\text{Parental Education}+0.507\cdot\text{Study Time Weekly}-2.775\cdot\text{Absences}+0.813\cdot\text{Tutoring}+0.403\cdot\text{Parental Support}+0.607\cdot\text{Extracurricular Activities}+0.399\cdot\text{Sports}+0.251\cdot\text{Music}-0.147\cdot\text{Volunteering}$$

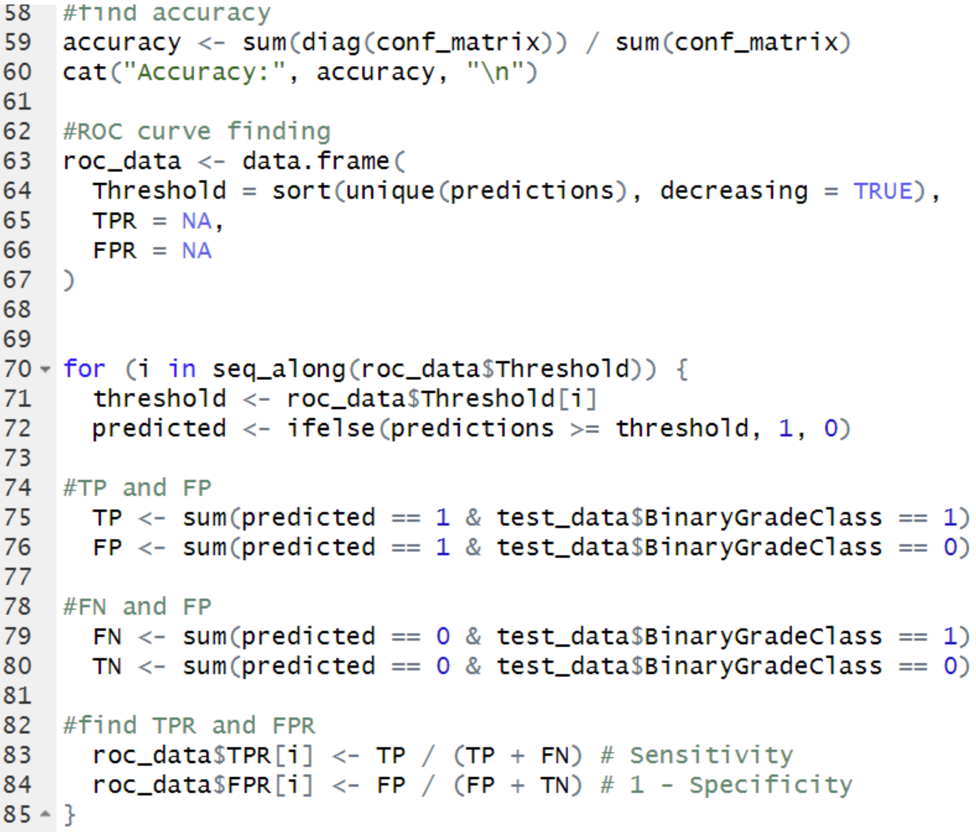
R STUDIO CODE:



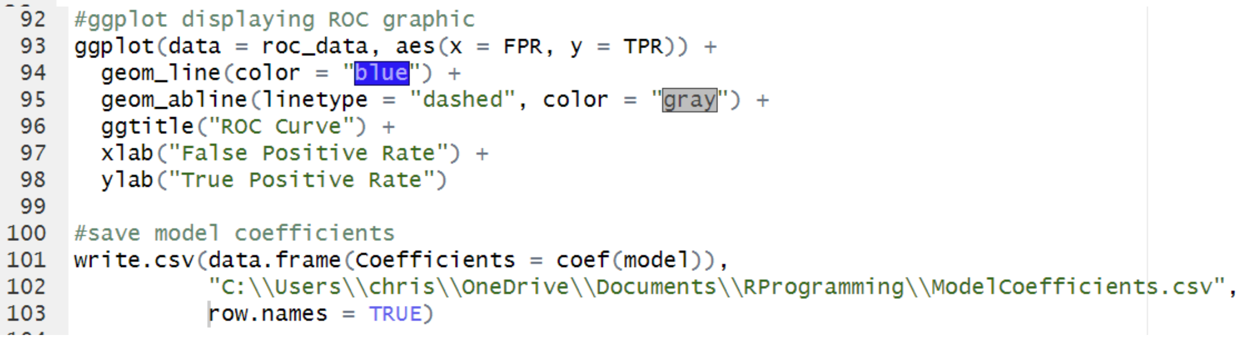
```r
school_data <- read.csv("C:\\Users\\chris\\OneDrive\\Documents\\RProgramming\\StudentPerformance.csv")
library(ggplot2)

#create new column
school_data$BinaryGradeClass <- ifelse(school_data$GradeClass <= 2, 1, 0)

#select coefficient features
features <- c("Age", "Gender", "Ethnicity", "ParentalEducation",
              "StudyTimeWeekly", "Absences", "Tutoring",
              "ParentalSupport", "Extracurricular",
              "Sports", "Music", "Volunteering")




school_data



#training and testing set split
set.seed(42)
train_indices <- sample(seq_len(nrow(school_data)), size = 0.7 * nrow(school_data))
train_data <- school_data[train_indices, ]
test_data <- school_data[-train_indices, ]



```



```r
#make sure that the numerics are the same in each column
numeric_features <- c("Age", "StudyTimeWeekly", "Absences")
for (feature in numeric_features) {
  mean_value <- mean(train_data[[feature]])
  sd_value <- sd(train_data[[feature]])
  train_data[[feature]] <- (train_data[[feature]] - mean_value) / sd_value
  test_data[[feature]] <- (test_data[[feature]] - mean_value) / sd_value
}

#logistic model fit
model <- glm(BinaryGradeClass ~ ., data = train_data[, c(features, "BinaryGradeClass")],
             family = "binomial")


summary(model)

#predict the test set
predictions <- predict(model, newdata = test_data[, features], type = "response")

#probabilities to binary outputs
threshold <- 0.5
predicted_classes <- ifelse(predictions > threshold, 1, 0)

#conf matrix
conf_matrix <- table(Actual = test_data$BinaryGradeClass, Predicted = predicted_classes)
print(conf_matrix)
```



```r
#find accuracy
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Accuracy:", accuracy, "\n")

#ROC curve finding
roc_data <- data.frame(
  Threshold = sort(unique(predictions), decreasing = TRUE),
  TPR = NA,
  FPR = NA
)


for (i in seq_along(roc_data$Threshold)) {
  threshold <- roc_data$Threshold[i]
  predicted <- ifelse(predictions >= threshold, 1, 0)

#TP and FP
  TP <- sum(predicted == 1 & test_data$BinaryGradeClass == 1)
  FP <- sum(predicted == 1 & test_data$BinaryGradeClass == 0)

#FN and FP
  FN <- sum(predicted == 0 & test_data$BinaryGradeClass == 1)
  TN <- sum(predicted == 0 & test_data$BinaryGradeClass == 0)

#find TPR and FPR
  roc_data$TPR[i] <- TP / (TP + FN) # Sensitivity
  roc_data$FPR[i] <- FP / (FP + TN) # 1 - Specificity
}
```



```r
#ggplot displaying ROC graphic
ggplot(data = roc_data, aes(x = FPR, y = TPR)) +
  geom_line(color = "blue") +
  geom_abline(linetype = "dashed", color = "gray") +
  ggtitle("ROC Curve") +
  xlab("False Positive Rate") +
  ylab("True Positive Rate")

#save model coefficients
write.csv(data.frame(Coefficients = coef(model)),
          "C:\\Users\\chris\\OneDrive\\Documents\\RProgramming\\ModelCoefficients.csv",
          row.names = TRUE)
```