# METHODS IN DATA SCIENCE FINAL PROJECT: STUDENTS PERFORMANCE

By Nina Peak

# BRIEF SUMMARY AND OBJECTIVES:

The aim of this data science research project is to conduct a comprehensive analysis and I've chosen a student performance dataset. This dataset encompasses various factors influencing students' academic achievements, including study course, previous grades, and demographic characteristics. Through statistical analysis and advanced visualization techniques, this research seeks to uncover patterns, trends, and correlations within the data. The insights gained will be invaluable for understanding the factors that impact student performance and for developing strategies to enhance academic outcomes.

# RESEARCH QUESTION AND SIGNIFICANCE

What is the correlation between students' past test scores, patterns involving parents' level of education, lunch type, and test preparation courses. And how can learning how these factors predict student test scores and future performance?

Investigating how relationships between different subject test scores, parental education levels, lunch type (e.g., standard vs. free/reduced), and test preparation courses affect student performance and can provide insights into the impacts on students' education. This understanding can help identify at-risk students who may need additional resources or support. Developing a predictive model based on these variables could help schools in identifying students who are likely to struggle. Early identification allows for more time to create better learning environments, which can improve overall academic performance.

# DATASET DETAILS

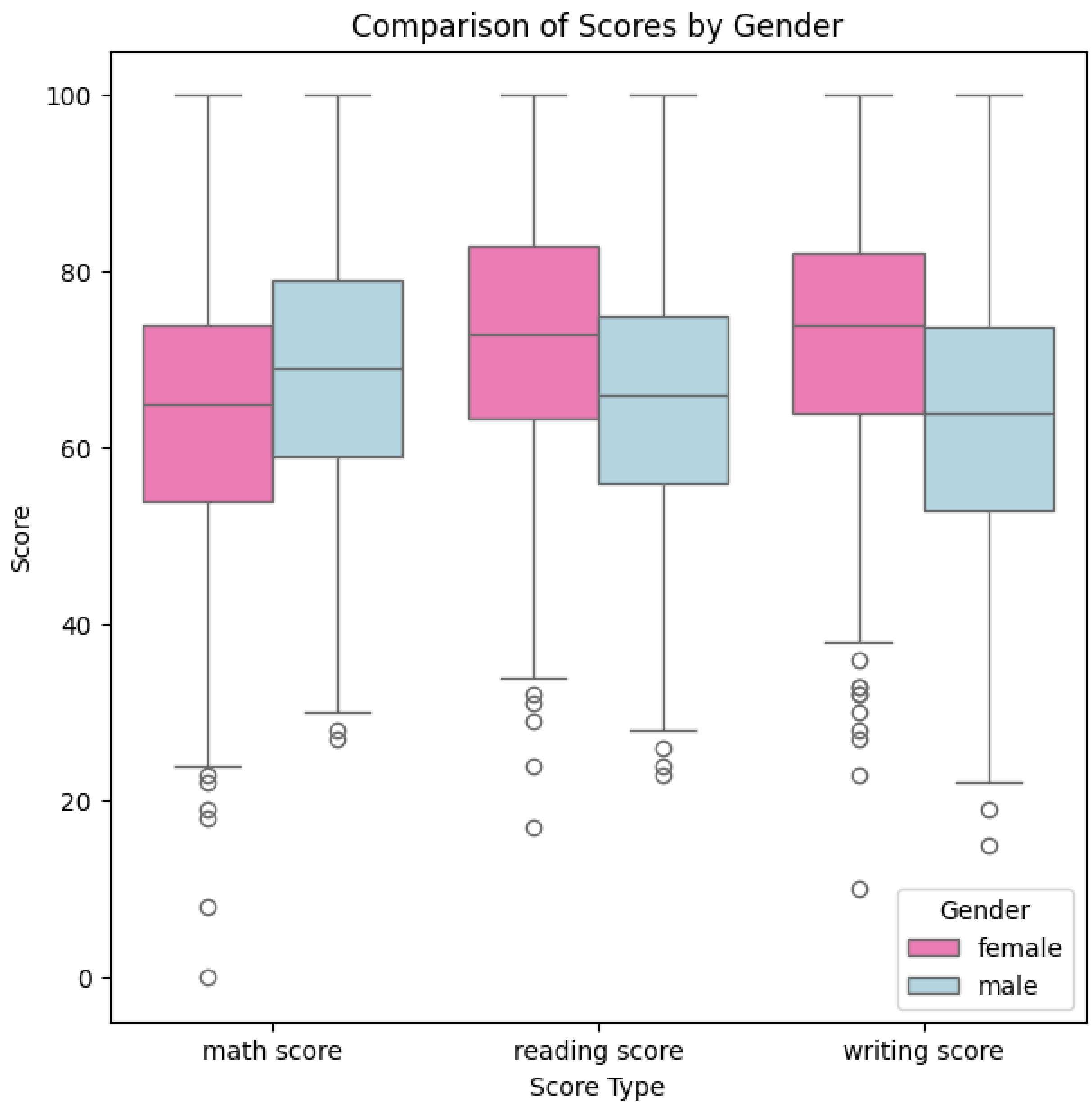
- Numerical and categorical data of 1,000 students
- Loading dataset: pd.read_csv('StudentsPerformance[1].csv')
- Head of the dataset:

| | gender | race/ethnicity | parental level of education | lunch | test preparation course | math score | reading score | writing score |
|---|---|---|---|---|---|---|---|---|
| 0 | female | group B | bachelor's degree | standard | none | 72 | 72 | 74 |
| 1 | female | group C | some college | standard | completed | 69 | 90 | 88 |
| 2 | female | group B | master's degree | standard | none | 90 | 95 | 93 |
| 3 | male | group A | associate's degree | free/reduced | none | 47 | 57 | 44 |
| 4 | male | group C | some college | standard | none | 76 | 78 | 75 |

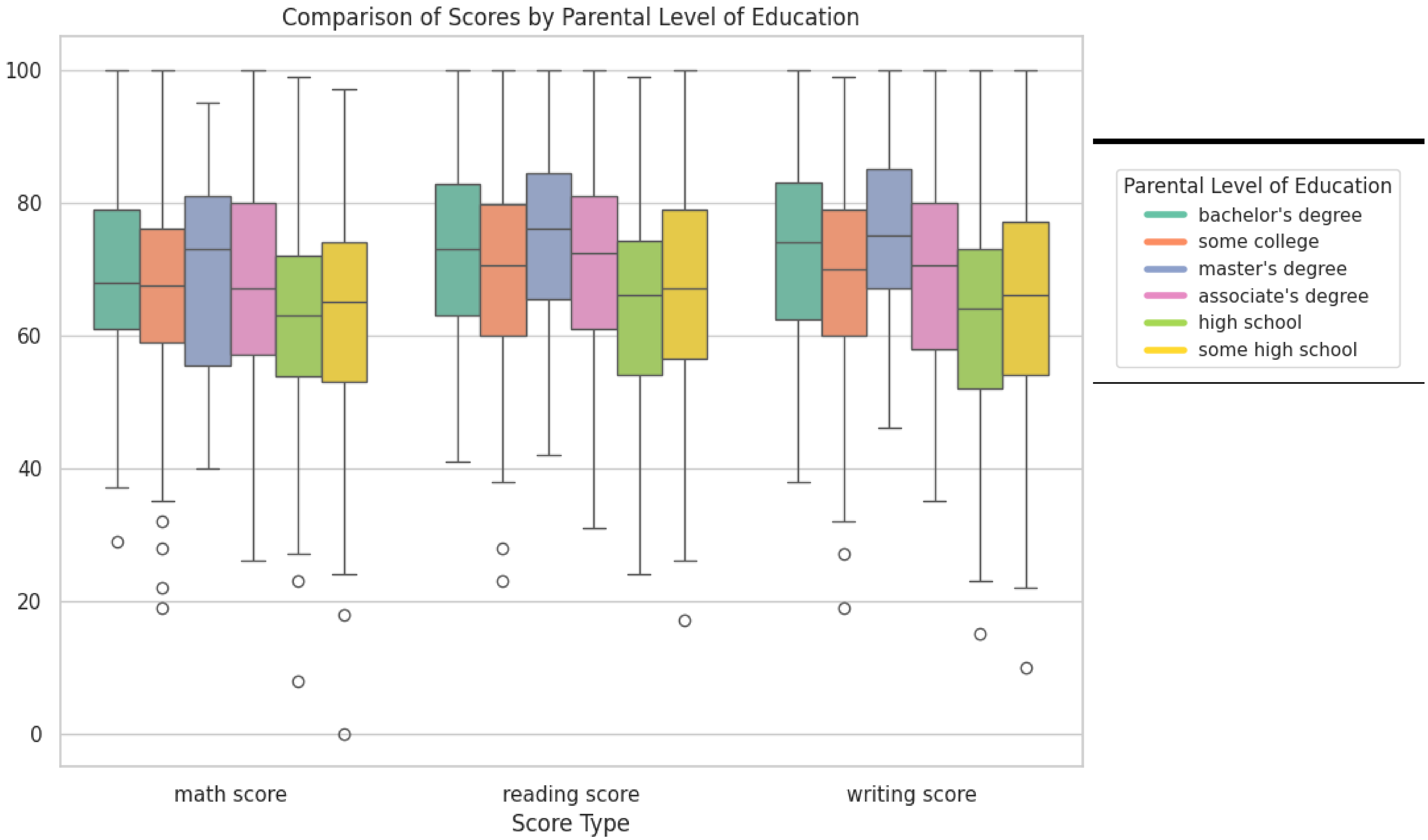
# CONFIGURATIONS USED

- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt
- import seaborn as sns

- from sklearn.datasets import make_classification
- from sklearn.linear_model import LogisticRegression
- from sklearn.metrics import accuracy_score, classification_report, mean_squared_error, r2_score
- from sklearn.model_selection import train_test_split, KFold, cross_val_score
- from sklearn.pipeline import make_pipeline
- from sklearn.preprocessing import LabelEncoder, StandardScaler
- from sklearn.model_selection import cross_val_score
- from sklearn.datasets import make_classification
- from sklearn.pipeline import make_pipeline




```
Statistics for Writing, Math, and Reading Scores by Gender:
        count       mean        std   min   25%   50%    75%    max
gender
female  518.0  72.467181  14.844842  10.0  64.0  74.0  82.00  100.0
male    482.0  63.311203  14.113832  15.0  53.0  64.0  73.75  100.0
        count       mean        std   min   25%   50%   75%    max
gender
female  518.0  63.633205  15.491453   0.0  54.0  65.0  74.0  100.0
male    482.0  68.728216  14.356277  27.0  59.0  69.0  79.0  100.0
        count       mean        std   min    25%   50%   75%    max
gender
female  518.0  72.608108  14.378245  17.0  63.25  73.0  83.0  100.0
male    482.0  65.473029  13.931832  23.0  56.00  66.0  75.0  100.0
```



Comparison of Scores by Parental Level of Education
100
80
60
40
20
0
math score
reading score
writing score
Score Type
Parental Level of Education
bachelor's degree
some college
master's degree
associate's degree
high school
some high school

# LINEAR REGRESSION

```python
import statsmodels.api as sm
# Encode categorical variables
spdf['lunch_encoded'] = spdf['lunch'].map({'standard': 0, 'free/reduced': 1})
spdf['gender_encoded'] = spdf['gender'].map({'female': 0, 'male': 1})

# Features and target variable
X = spdf[['math score', 'reading score', 'lunch_encoded', 'gender_encoded']]
y = spdf['writing score']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)
```

```python
# Create and fit the model using sklearn
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Display coefficients and intercept
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)

# Now use statsmodels for detailed summary
X_train_sm = sm.add_constant(X_train)  # Add a const
sm_model = sm.OLS(y_train, X_train_sm).fit()  # Fit

# Print the summary
print(sm_model.summary())
```

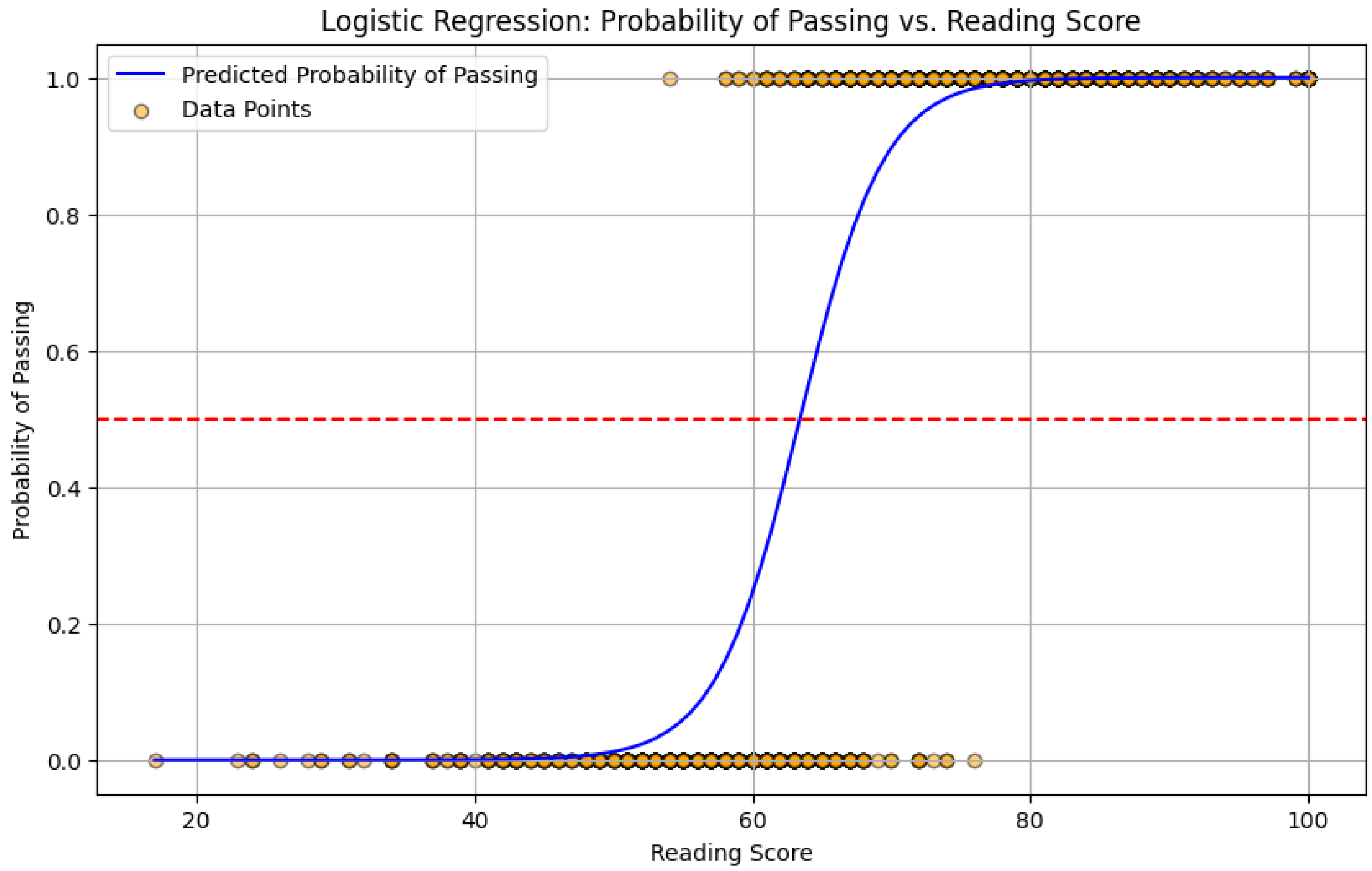
```
Coefficients: [ 0.25470267  0.73186269  0.09644226 -5.28642969]
Intercept: 3.026971099654986
                            OLS Regression Results
==============================================================================
Dep. Variable:          writing score   R-squared:                       0.928
Model:                            OLS   Adj. R-squared:                  0.928
Method:                 Least Squares   F-statistic:                     2573.
Date:                Thu, 05 Dec 2024   Prob (F-statistic):               0.00
Time:                        20:40:23   Log-Likelihood:                -2255.6
No. Observations:                 800   AIC:                             4521.
Df Residuals:                     795   BIC:                             4545.
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
const              3.0270      0.818      3.702      0.000       1.422       4.632
math score         0.2547      0.024     10.784      0.000       0.208       0.301
reading score      0.7319      0.024     30.447      0.000       0.685       0.779
lunch_encoded      0.0964      0.326      0.296      0.768      -0.544       0.737
gender_encoded    -5.2864      0.400    -13.211      0.000      -6.072      -4.501
==============================================================================
Omnibus:                        4.212   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.122   Jarque-Bera (JB):                4.051
Skew:                           0.134   Prob(JB):                        0.132
...
==============================================================================
```


Math Score vs. Writing Score
writing score
math score
0
20
40
60
80
100
Reading Score vs. Writing Score
reading score

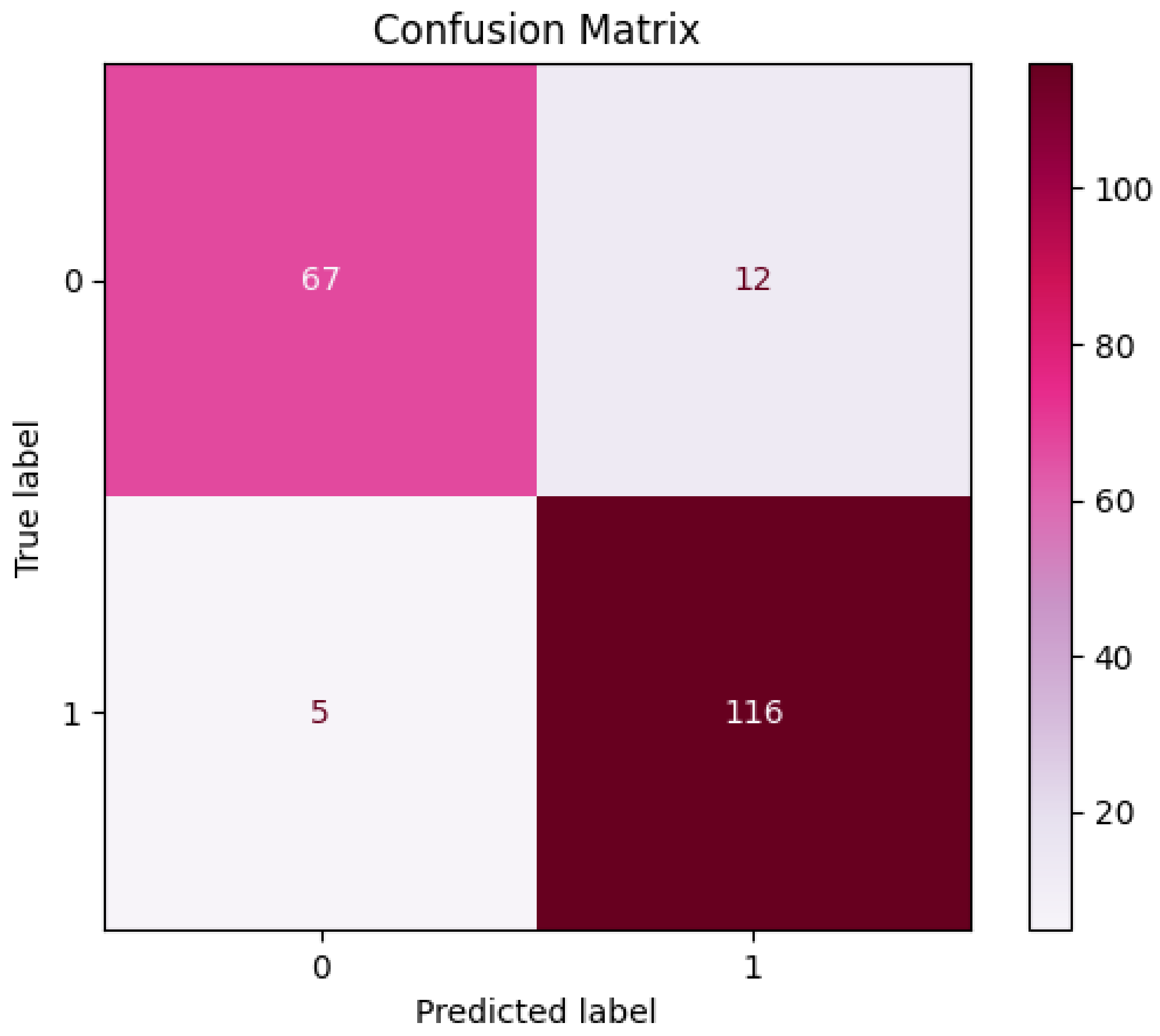
# CLASSIFICATION- LOGISTIC REGRESSION

```python
# Create binary target variable
spdf['pass'] = (spdf['writing score'] >= 65).astype(int)

# Features and target variable
X_class = spdf[['math score', 'reading score', 'lunch_encoded', 'gender_encoded']]
y_class = spdf['pass']

# Split the data into training and testing sets
X_train_class, X_test_class, y_train_class, y_test_class = train_test_split(X_class

# Fit the logistic regression model using statsmodels
X_train_class_sm = sm.add_constant(X_train_class)  # Add a constant
logit_model = sm.Logit(y_train_class, X_train_class_sm)
result = logit_model.fit()

# Print the summary
print(result.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                   pass   No. Observations:                  800
Model:                          Logit   Df Residuals:                      795
Method:                           MLE   Df Model:                            4
Date:                Thu, 05 Dec 2024   Pseudo R-squ.:                  0.7275
Time:                        20:40:23   Log-Likelihood:                -145.18
converged:                       True   LL-Null:                       -532.70
Covariance Type:            nonrobust   LLR p-value:                1.970e-166
==================================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------
const            -26.3175      2.363    -11.137      0.000     -30.949     -21.686
math score         0.0989      0.024      4.121      0.000       0.052       0.146
reading score      0.3299      0.036      9.112      0.000       0.259       0.401
lunch_encoded     -0.1522      0.317     -0.481      0.631      -0.773       0.468
gender_encoded    -2.2698      0.437     -5.192      0.000      -3.127      -1.413
==================================================================================
```




```python
reading_scores = np.linspace(spdf['reading score'].min(), spdf['reading score'].max(), 100)

# Calculate the predicted probabilities for different reading scores while holding other variables constant
# Let's assume average values for math score and lunch_encoded, gender_encoded
avg_math = spdf['math score'].mean()
avg_lunch = spdf['lunch_encoded'].mean()
avg_gender = spdf['gender_encoded'].mean()

# Calculate log odds
log_odds = (result.params['const'] +
            result.params['math score'] * avg_math +
            result.params['lunch_encoded'] * avg_lunch +
            result.params['gender_encoded'] * avg_gender +
            result.params['reading score'] * reading_scores)

# Convert log odds to probabilities
predicted_probabilities = 1 / (1 + np.exp(-log_odds))

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(reading_scores, predicted_probabilities, label='Predicted Probability of Passing', color='blue')
plt.scatter(spdf['reading score'], spdf['pass'], alpha=0.5, label='Data Points', color='orange', edgecolor=
plt.title('Logistic Regression: Probability of Passing vs. Reading Score')
plt.xlabel('Reading Score')
plt.ylabel('Probability of Passing')
plt.axhline(0.5, linestyle='--', color='red')  # Add a line at 0.5 for the decision boundary
plt.legend()
plt.grid()
plt.show()
```




- True Positive (TP): 67
- This means that 67 students who actually passed were correctly predicted to pass by the model. These are the true positives.
- False Negative (FN): 12
- This indicates that 12 students who actually passed were incorrectly predicted to fail. These are the false negatives, meaning the model missed identifying these students as passers.
- False Positive (FP): 5
- This means that 5 students who actually failed were incorrectly predicted to pass. These are the false positives, indicating that the model incorrectly flagged these students as passers.
- True Negative (TN): 116
- This indicates that 116 students who actually failed were correctly predicted to fail. These are the true negatives.
- Test Accuracy: 90.5%

# REFERENCES


- Kaggle dataset- https://www.kaggle.com/datasets/rabieelkharoua/students-performance-dataset
- Class book- https://www.statlearning.com/
- Scikit learn- https://scikit-learn.org/stable/