Report

Group members -

- Basavachari Boppudi (S20200010043)
- Tummepalli Anka Chandrahas (S20200010213)

Outline -

- Problem statement
- Objective
- Dataset
- Methodology
- Code summary

Problem statement -

- Do Exploratory Data Analysis (EDA) for the students performance in exams dataset and build a model for classifying these students.

Objective -

- The objective of this project is to do EDA and try different models for the classification and choose the best model. We used Random Forest classifier, Decision tree, K-neighbors classifier and Logistic Regression.

Dataset -

- We have taken the **Students Performance in Exams** dataset from kaggle which is about the marks secured by the students. The attributes in the dataset are as follows
    - gender - Gender of the student (categorical -[male,female])
    - race/ethnicity - race of the student (categorical - [group A, group B, group C, group D, group E])
    - parental level of education - highest degree of parents (categorical - [bachelor's degree, some college, master's degree, associate's degree, high school, some high school])
    - lunch - lunch fee for the student (categorical - [standard, free/reduced])
    - test preparation course - students status whether a course has been taken and completed or none (categorical - [none, completed])
    - math score - student performance in math test (numerical)
    - reading score - student performance in reading test (numerical)
    - writing score - student performance in writing test (numerical)

- Data sample

	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

Methodology -

- Importing Data
- Data preparation
- Data preprocessing
    - Checking for null values
    - Checking datatypes of each attribute
    - Correlation between each attribute
- Data Visualization
- Model
    - Splitting train and test set using train_test_split
    - Checking evaluation metrics for Random Forest classifier, Decision tree, K-neighbors classifier and Logistic Regression to choose the best model for this dataset.
- Results

Code summary -

Importing data

- Importing dataset into a pandas dataframe.

	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

Data Preparation

- We added an attribute called ‘percentage’ to the dataset by taking the average of all 3 scores of a student.
- After adding ‘percentage’ attribute to the dataset,

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	Percentage
0	female	group B	bachelor's degree	standard	none	72	72	74	72.67
1	female	group C	some college	standard	completed	69	90	88	82.33
2	female	group B	master's degree	standard	none	90	95	93	92.67
3	male	group A	associate's degree	free/reduced	none	47	57	44	49.33
4	male	group C	some college	standard	none	76	78	75	76.33

Data Preprocessing

- We checked for null values in the dataset

```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
Percentage                     0
dtype: int64
```

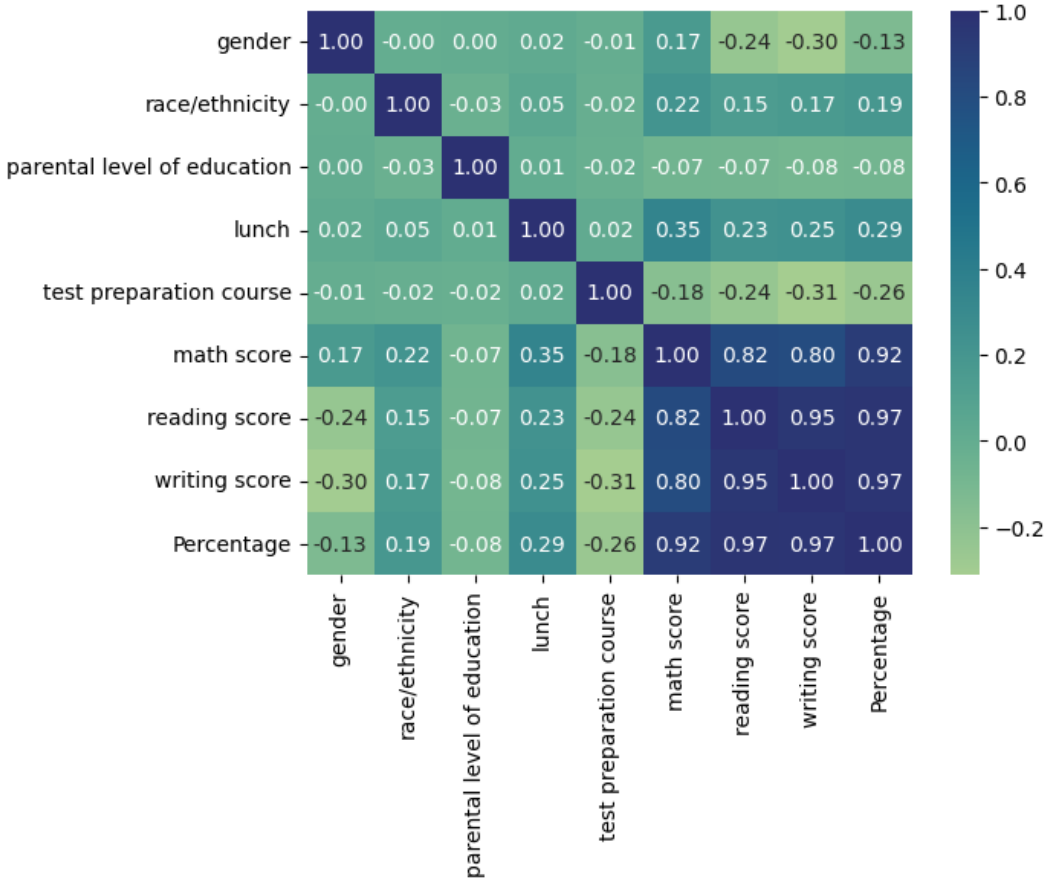
There are no null values in the dataset

- We check what are the datatypes of each attribute in the dataset

```
gender                          object
race/ethnicity                  object
parental level of education     object
lunch                           object
test preparation course         object
math score                       int64
reading score                    int64
writing score                    int64
Percentage                     float64
dtype: object
```

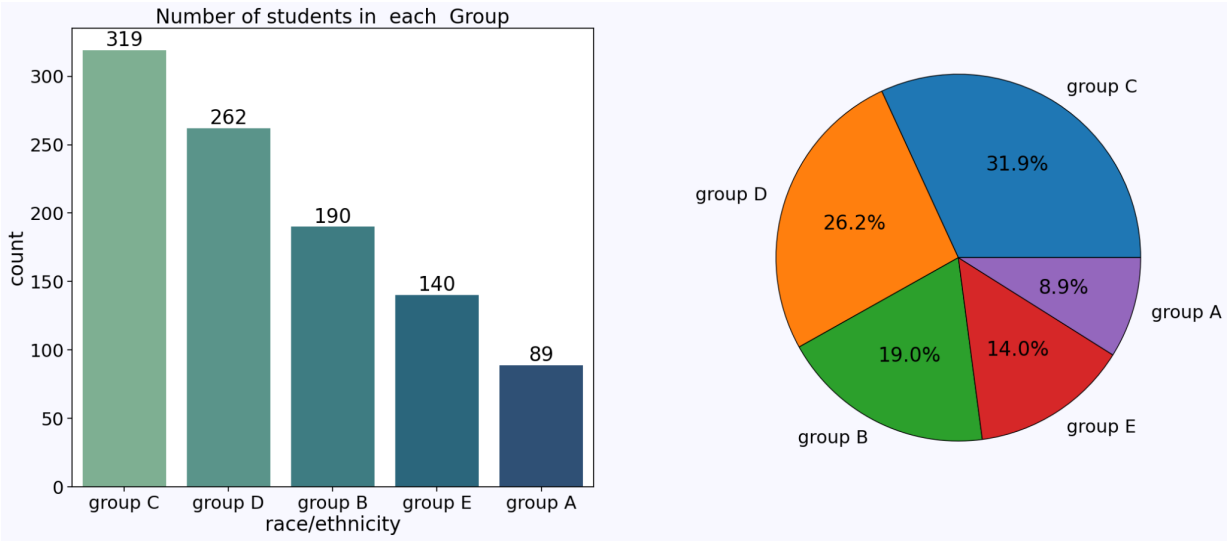
There are 5 categorical attributes and 4 numerical attributes in the dataset

- Checking for correlation among attributes. First we converted all the categorical attributes into numerical using the label encoder method in sklearn library. Heat map for correlation matrix is as follows

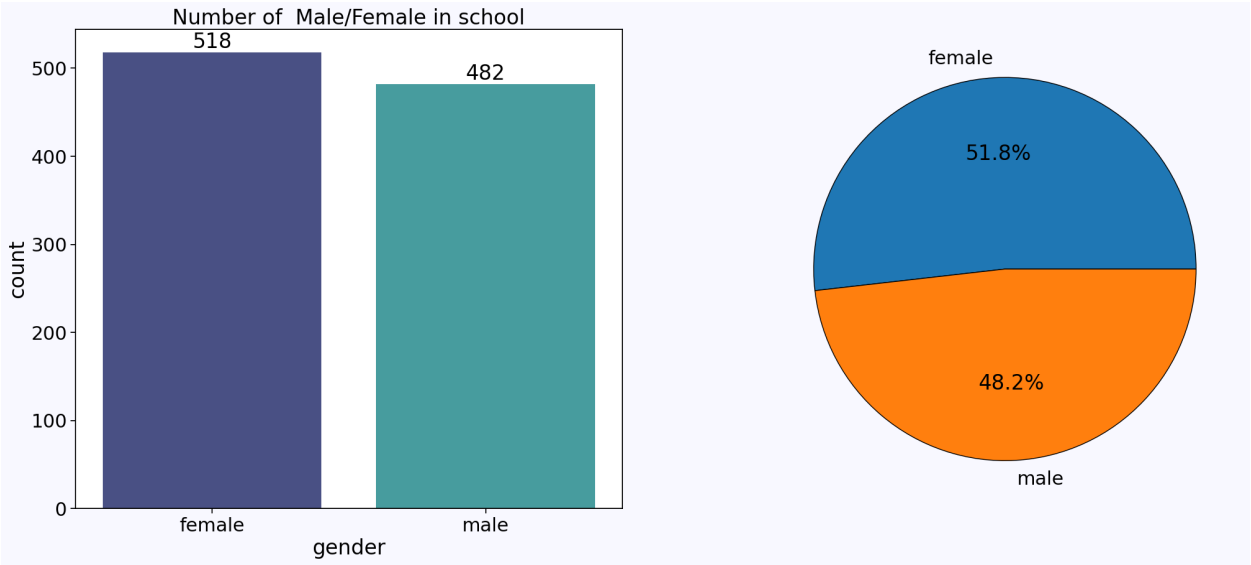


There is no correlation between independent attributes or independent and dependent attributes.

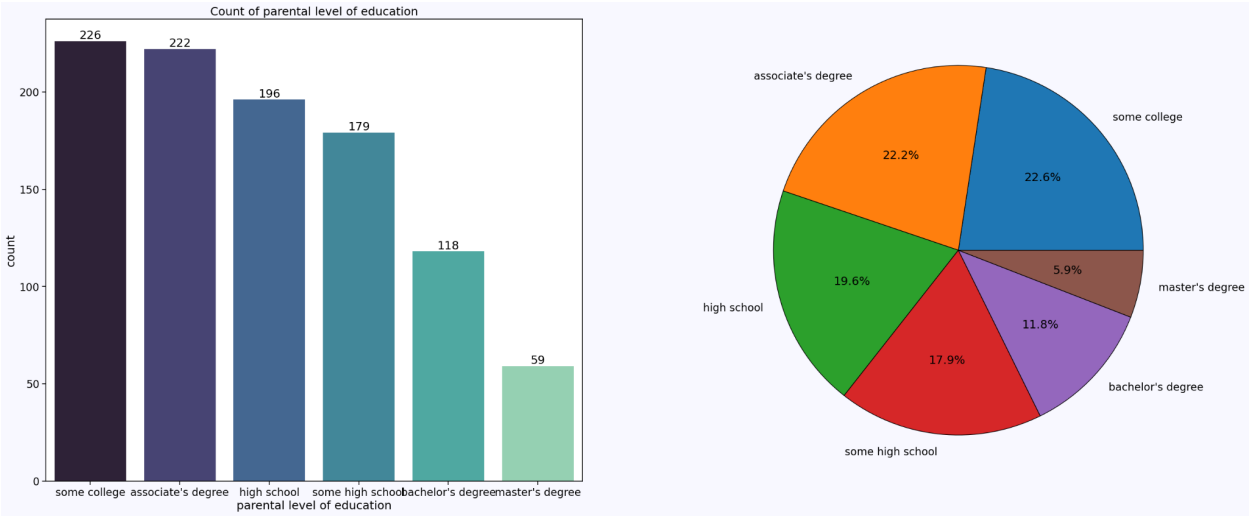
Data Visualization



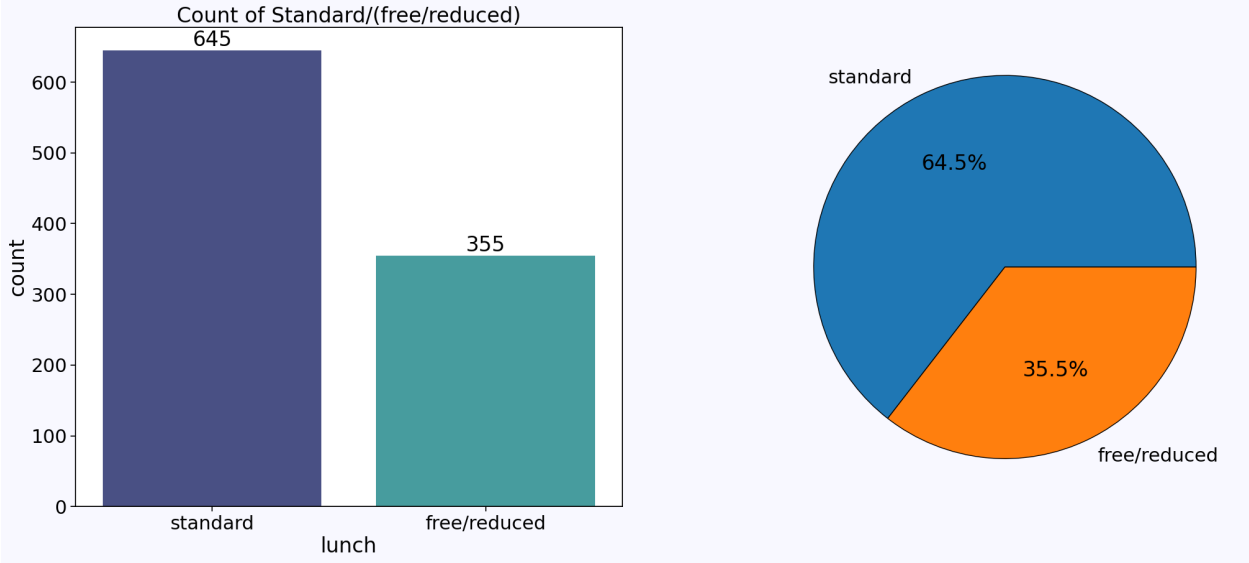
- This plot is about number of students in each group in the race/ethnicity attribute
- We can see that group C has the highest number of students and group A has the lowest



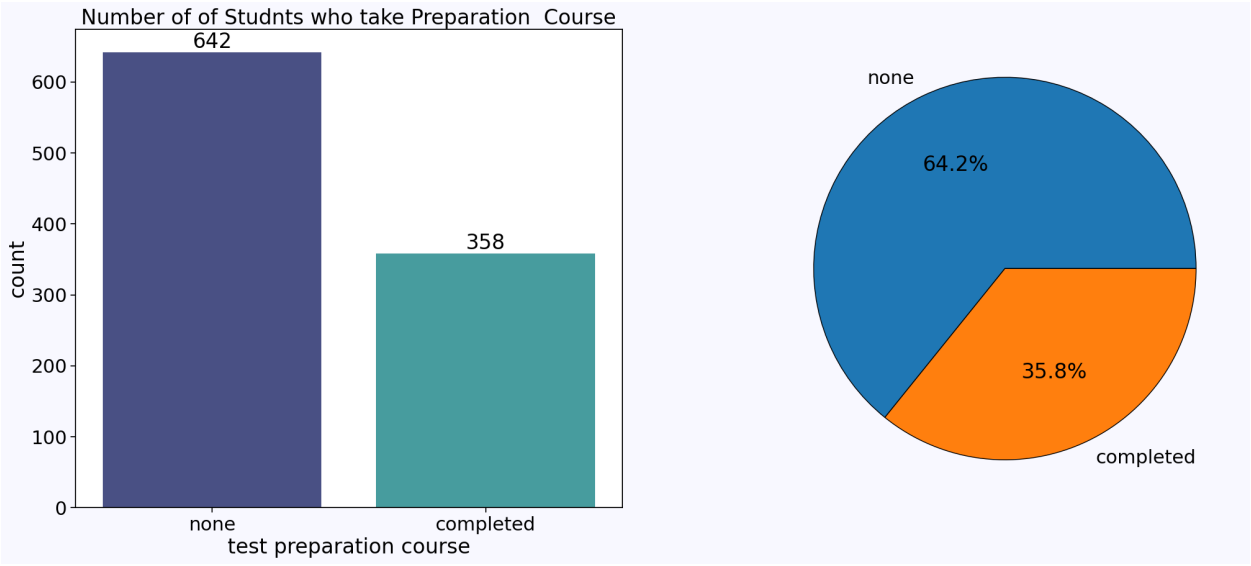
- This plot is about number of students in each gender in gender attribute
- Students are almost equally distributed between male and female.



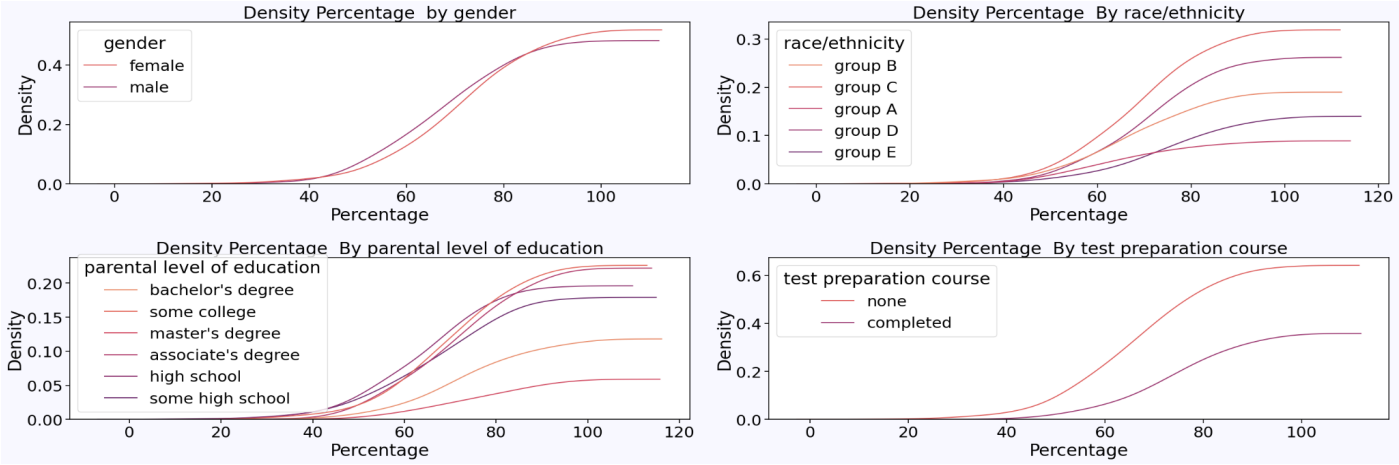
- This plot is about number of students in each parent's level of education
- Most parent's education is some college and associate degree is second highest with not much difference. Least is master's degree.



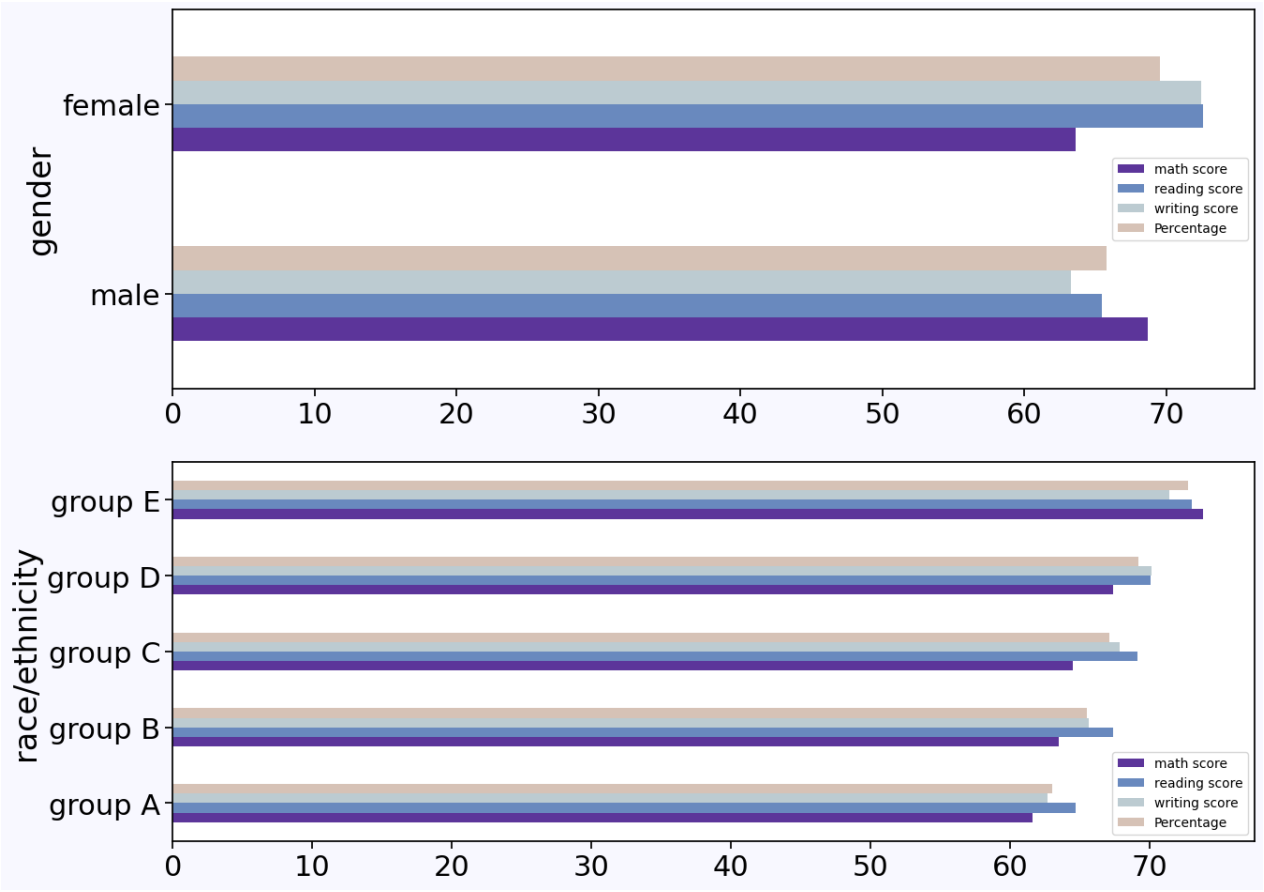
- This plot is about the number of students distributed in the lunch attribute.
- Most of the students(64.5%) are paying standard fees.



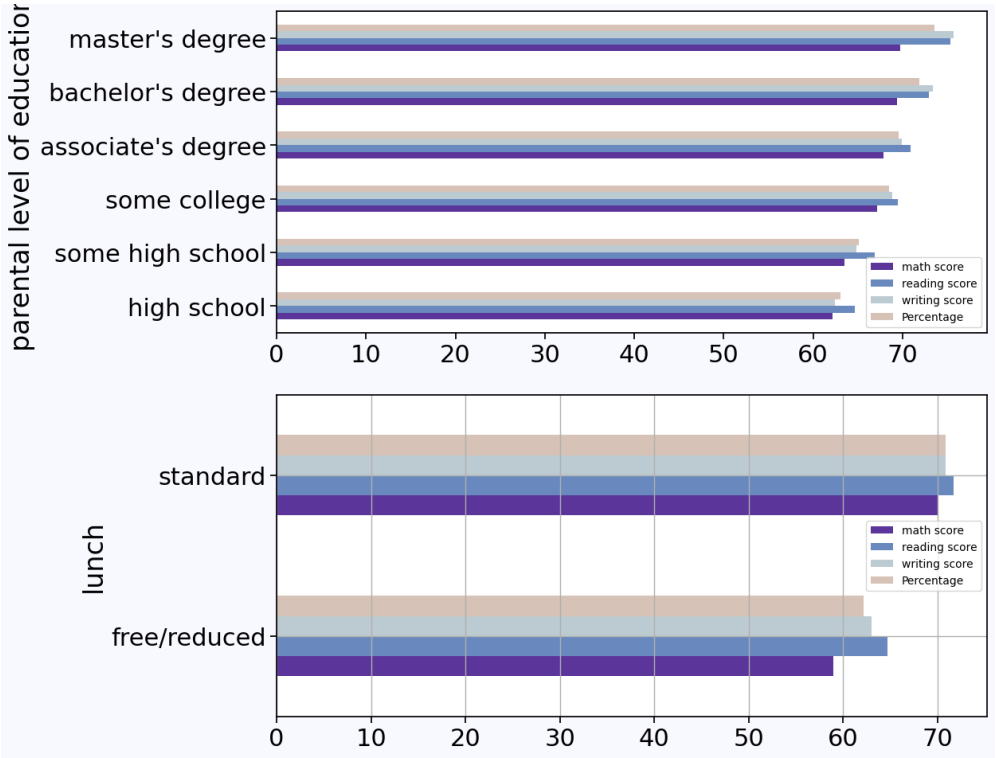
- This plot is about the number of students distributed in test preparation course attribute.
- Most of the students(64.2%) have not taken any test preparation course.



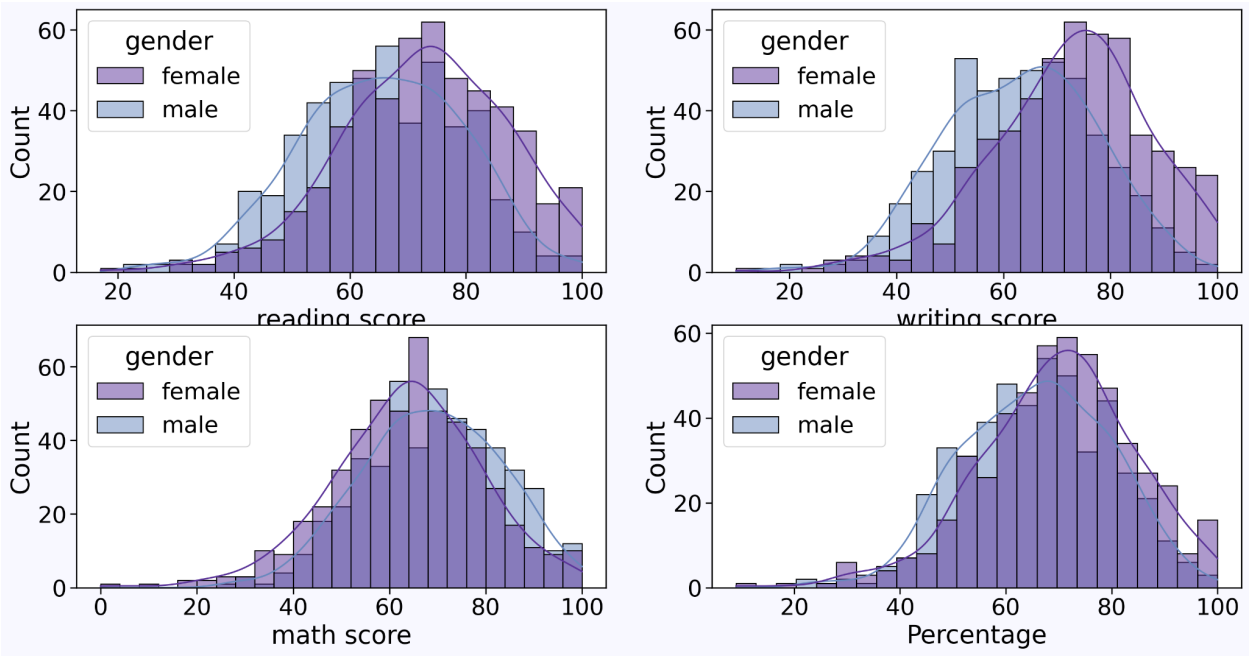
- This density-Percentage plot gives the probability distribution of percentage with various classes.



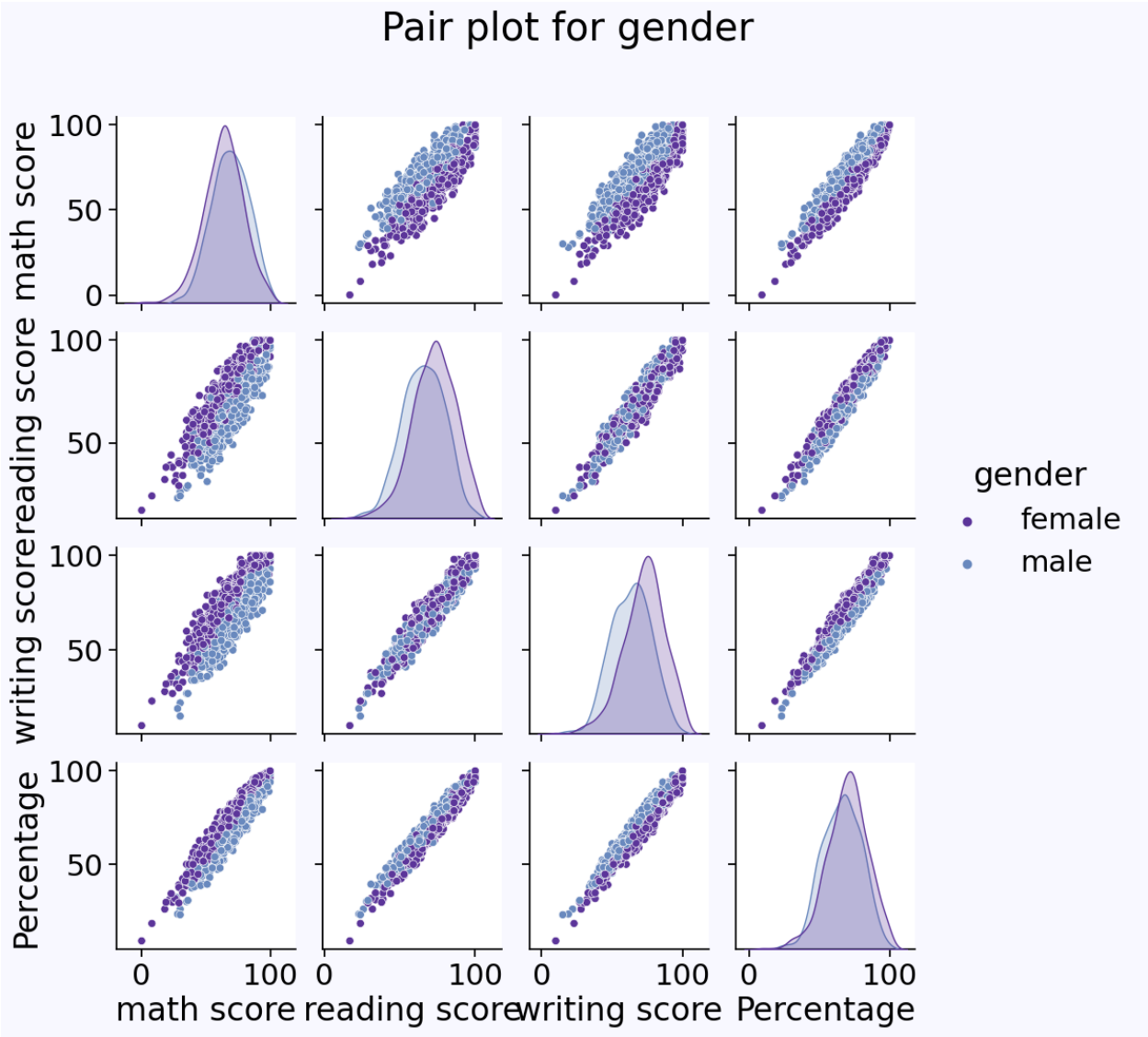
- These are the bar plots of the score vs gender and score vs race/ethnicity.
- Females have achieved highest in both reading and writing score but Males get the highest in Math score.
- Group E got the highest score in all 3 scores among the groups.



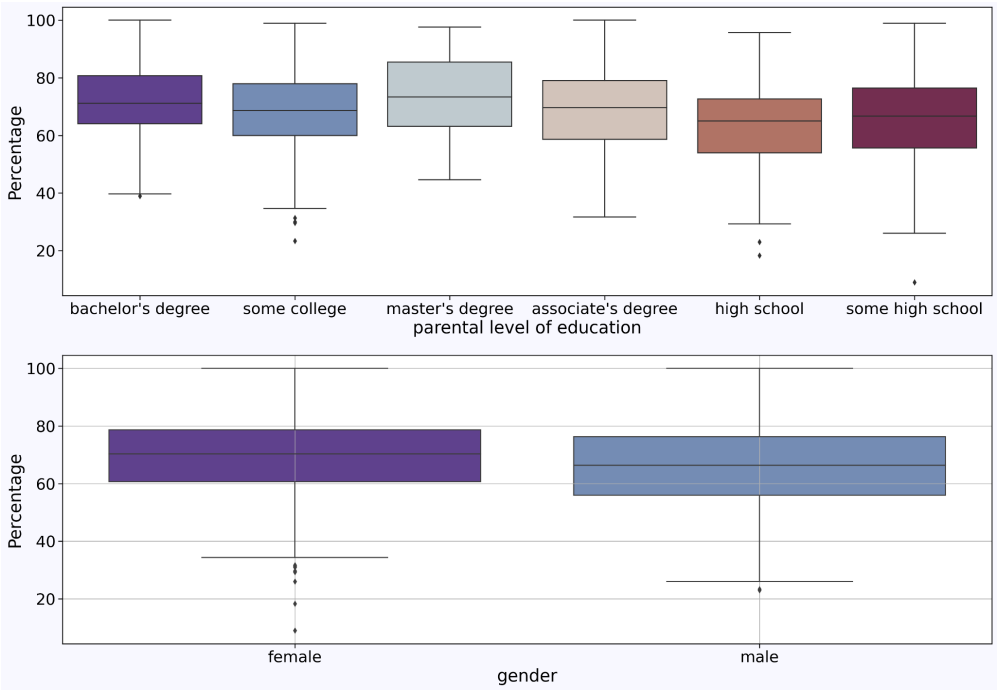
- Parental having of Master's degree students got the highest score than others
- Students taking Standard lunch got the highest score than free lunch.

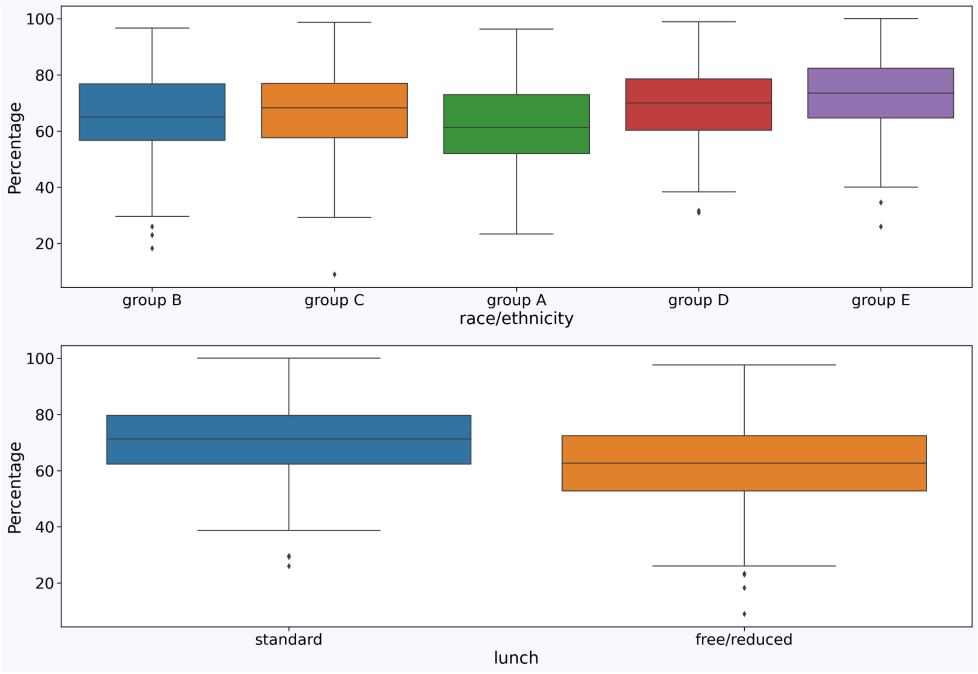


- Almost all the scores are in normal distribution for both genders.



- The pair plot all scores for each other.
- It states that all the scores are correlated positively as we can see the contour along the ‘x=y’ line.





- These are the box plot for parents level of education, gender, race/ethnicity and lunch attributes
- There are very few outliers in the dataset which are not really necessary to remove.

Model -

- These are the unique values present into the categorical attributes

```
gender unique values: ['female' 'male']

ethnicity unique values: ['group B' 'group C' 'group A' 'group D' 'group E']

parent_education unique values: ["bachelor's degree" 'some college' "master's degree" "associate's degree"
 'high school' 'some high school']

lunch unique values: ['standard' 'free/reduced']

prep_course unique values: ['none' 'completed']
```

- We added a new attribute called grade for classification using the following piece of code

```
def letter_grade(percentage):
    if percentage >= 90:
        return 'A'
    elif percentage < 90 and percentage >= 80:
        return 'B'
    elif percentage < 80 and percentage >= 70:
        return 'C'
    elif percentage < 70 and percentage >= 60:
        return 'D'
    else:
        return 'F'
```

After doing it the dataset is like

	gender	ethnicity	parent_education	lunch	prep_course	math_score	reading_score	writing_score	Percentage	grades
0	female	group B	bachelor's degree	standard	none	72	72	74	72.67	C
1	female	group C	some college	standard	completed	69	90	88	82.33	B
2	female	group B	master's degree	standard	none	90	95	93	92.67	A
3	male	group A	associate's degree	free/reduced	none	47	57	44	49.33	F
4	male	group C	some college	standard	none	76	78	75	76.33	C

Value counts of each grade are as shown →

```
F    285
C    261
D    256
B    146
A     52
Name: grades, dtype: int64
```

So, the target variable is grades.

- We did one more encoding scheme using the get_dummies method in the pandas library for this dataset to get understanding of different encoding methods.

	math_score	reading_score	writing_score	grades	gender_male	ethnicity_group B	ethnicity_group C	ethnicity_group D	ethnicity_group E	parent_education_bachelor's degree	parent_ed
0	72	72	74	3	0	1	0	0	0	1	
1	69	90	88	2	0	0	1	0	0	0	
2	90	95	93	1	0	1	0	0	0	0	
3	47	57	44	5	1	0	0	0	0	0	
4	76	78	75	3	1	0	1	0	0	0	

- We defined a custom transformer named student_transform which is like a preprocessing step for model pipeline. We used 4 different ML models which are Random Forest classifier, Decision tree, K-neighbors classifier, and Logistic Regression.
- We did train_test_split with test size as 20%

```
class Student_Transformer(BaseEstimator, TransformerMixin):
    def __init__(self):
        return None
    def transform(self, X):
        X = X
        X = X.drop(['grades'], axis = 1)
        return X
    def fit(self, X, y):
        return self

clfs = [RandomForestClassifier(n_estimators=50, random_state=42),
        DecisionTreeClassifier(random_state=42),
        KNeighborsClassifier(n_neighbors=5),
        LogisticRegression(max_iter=80, random_state=42)]

X_train, X_test, y_train, y_test = train_test_split(class_df, class_df.grades.ravel(), test_size=0.20, random_state=42)
```

- For the model we created a pipeline which includes 3 steps
    - Student_Transformer - This is a preprocessing step which removes grades attribute.
    - StandardScalar - This step is to normalize the data using z - score normalization method.
    - Classifier - This step is for training of the 4 models which are mentioned above.

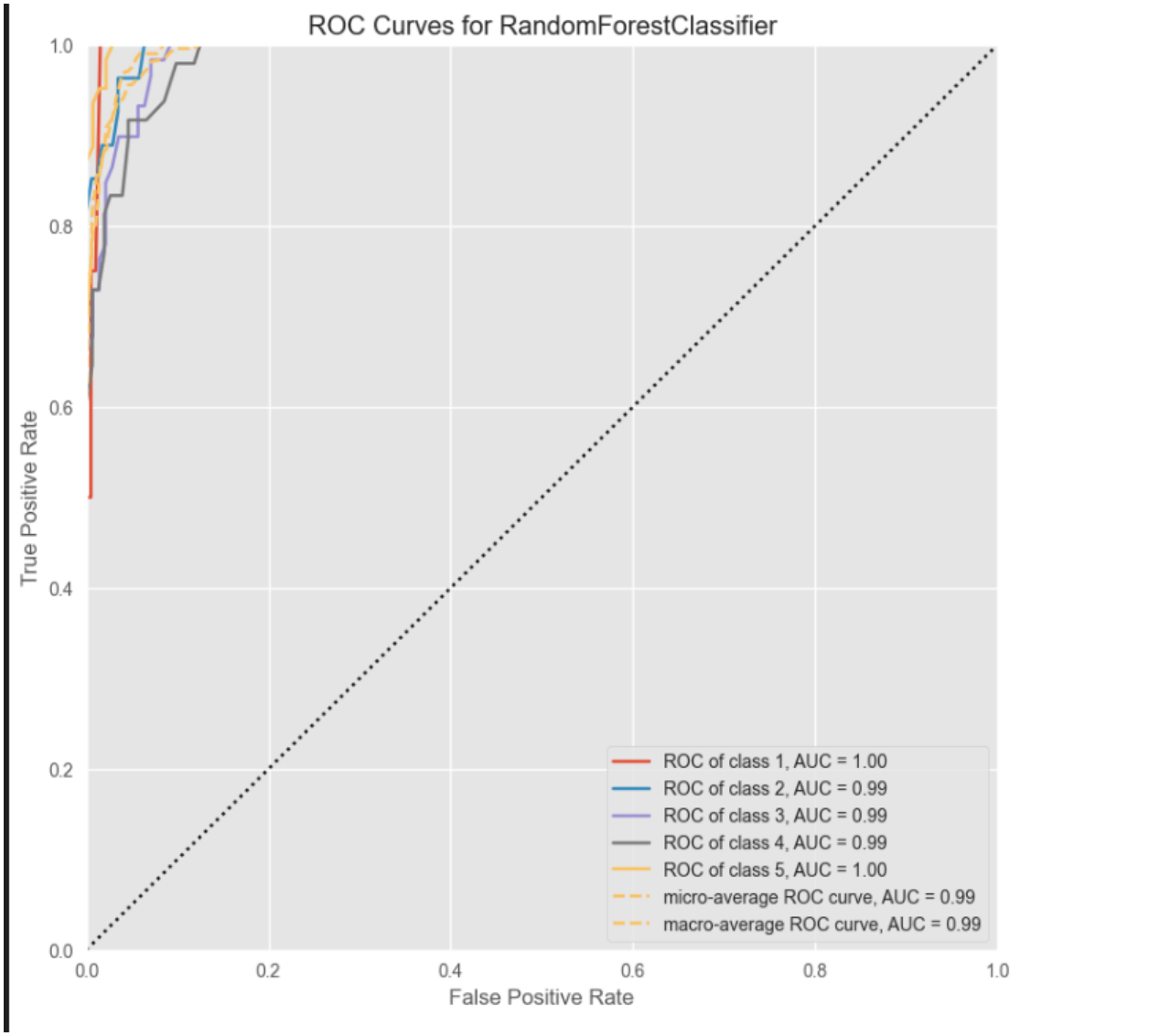
```
def model():
    fig = plt.figure(figsize=(20,20))
    k = 1
    for x in clfs:
        pipe = Pipeline([
            ('student', Student_Transformer()),
            ('std', StandardScaler(),),
            ('class', x)
        ])

        pipe.fit(X_train, y_train)
        classifier = pipe['class'].__class__.__name__
        print("-"*40,classifier,"-"*40, end="\n\n")
        y_pred = pipe.predict(X_test)
        cm = confusion_matrix(y_test, y_pred)
        print(f"The confusion matrix for {classifier}:")
        print(cm)
        bal_acc = balanced_accuracy_score(y_test, y_pred)
        acc = accuracy_score(y_test, y_pred)
        sen = sensitivity_score(y_test, y_pred, average='weighted')
        spec = specificity_score(y_test, y_pred, average='weighted')
        print(f"The accuracy score for {classifier} is {round((acc*100),2)}%")
        print(f"The balanced accuracy score for {classifier} is {round((bal_acc*100),2)}%")
        print(f"The sensitivity for {classifier} is {round((sen*100),2)}%")
        print(f"The specificity for {classifier} is {round((spec*100),2)}%")
        with plt.style.context("ggplot"):
            plt.subplot(2,2,k)
            roc = ROCAUC(pipe)
            roc.fit(X_train, y_train)
            roc.score(X_test, y_test)
            k += 1
            roc.finalize()
            plt.grid(True)
    plt.show()
```

- Evaluation metrics used are - confusion matrix, accuracy, balanced accuracy, sensitivity, specificity and ROC curve.

Results

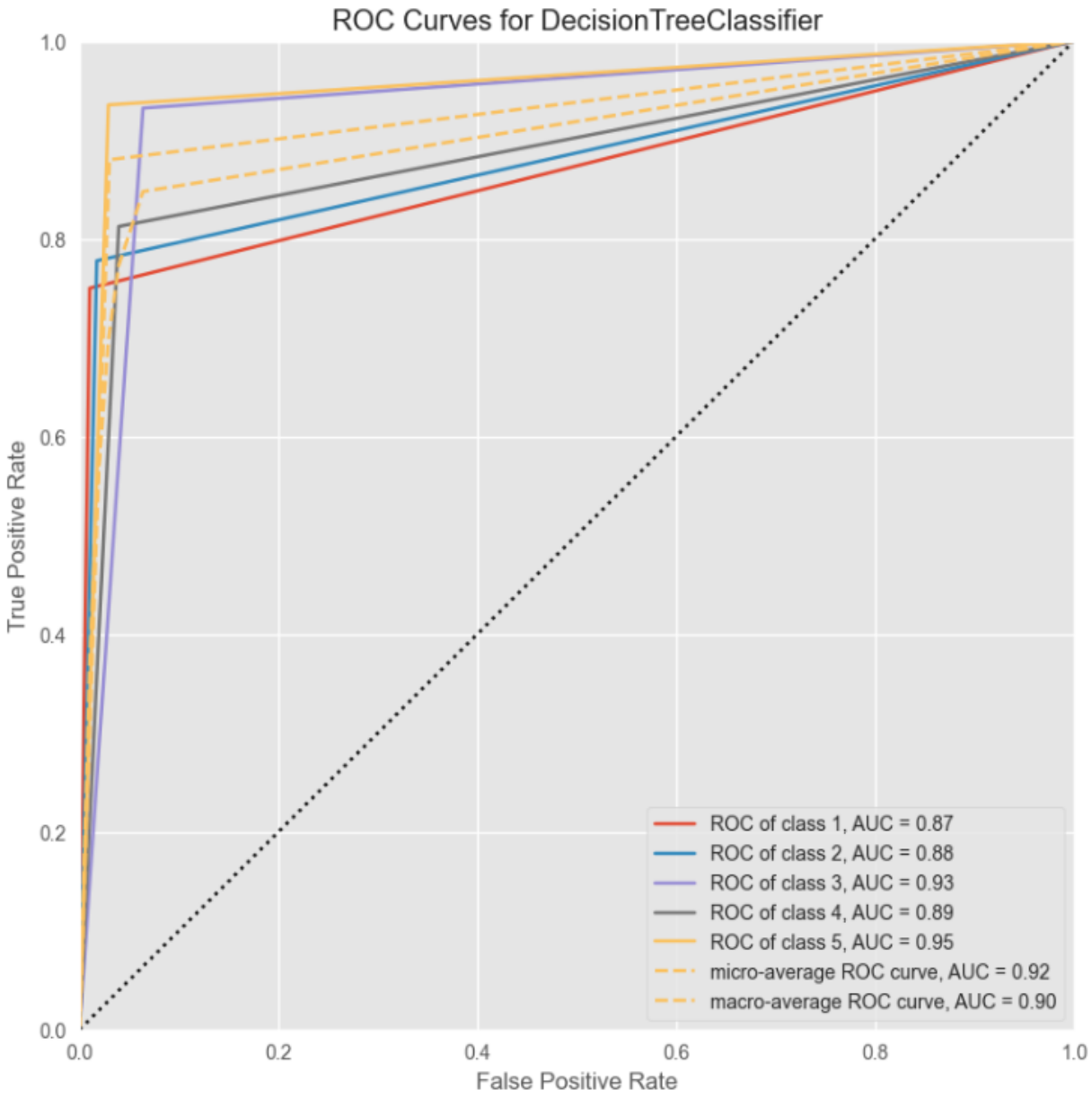
- Evaluation results for random forest classifier

```
---------------------------------------- RandomForestClassifier ----------------------------------------

The confusion matrix for RandomForestClassifier:
[[ 2  2  0  0  0]
 [ 1 23  3  0  0]
 [ 0  1 55  3  0]
 [ 0  0  5 40  3]
 [ 0  0  0  2 60]]
The accuracy score for RandomForestClassifier is 90.0%
The balanced accuracy score for RandomForestClassifier is 81.7%
The sensitivity for RandomForestClassifier is 90.0%
The specificity for RandomForestClassifier is 96.62%
```



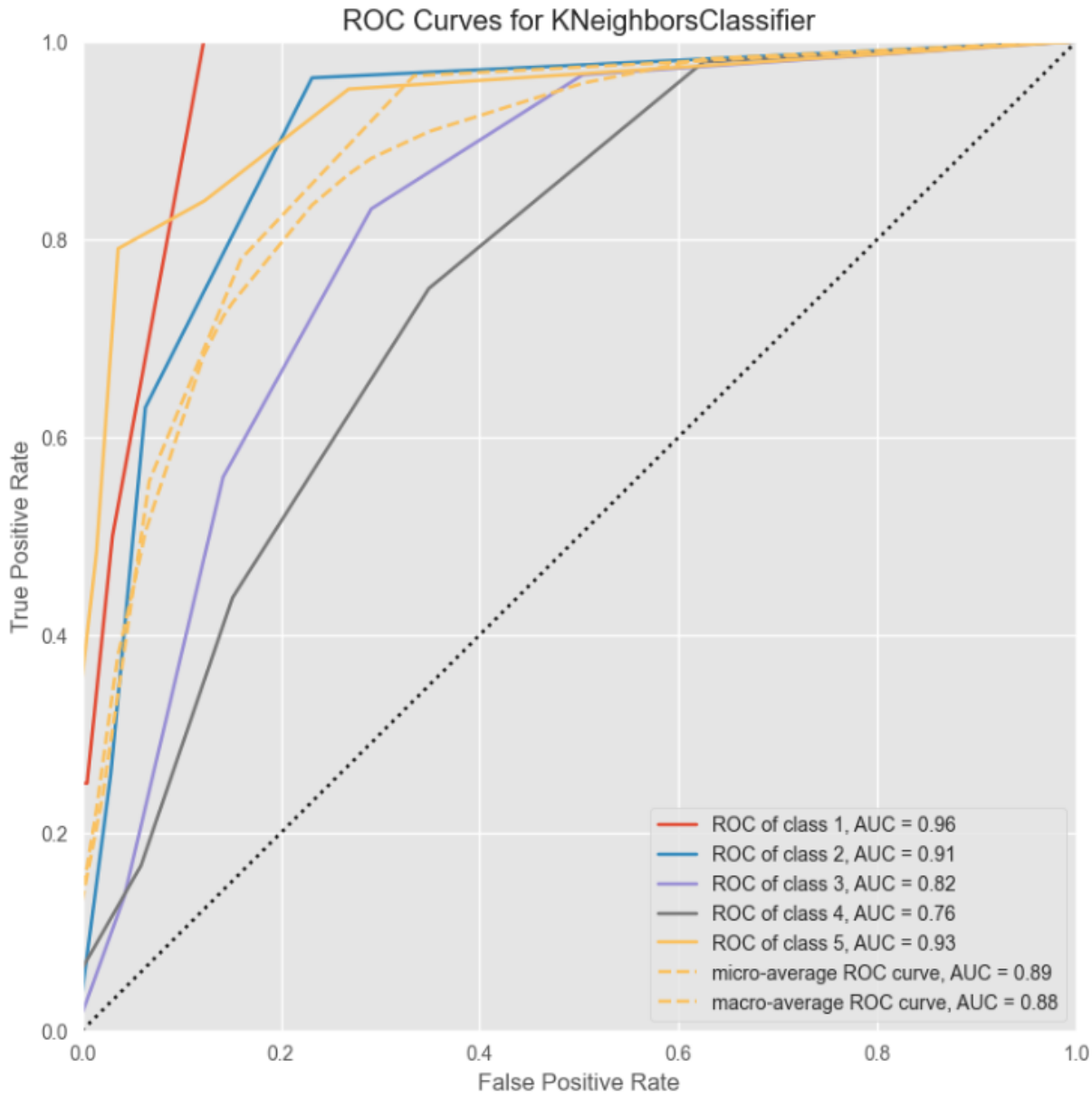
- Evaluation results for *Decision tree* classifier

```
The specificity for RandomForestClassifier is 96.02%
-------------------------------------- DecisionTreeClassifier --------------------------------------

The confusion matrix for DecisionTreeClassifier:
[[ 3  1  0  0  0]
 [ 2 21  4  0  0]
 [ 0  2 55  2  0]
 [ 0  0  5 39  4]
 [ 0  0  0  4 58]]
The accuracy score for DecisionTreeClassifier is 88.0%
The balanced accuracy score for DecisionTreeClassifier is 84.16%
The sensitivity for DecisionTreeClassifier is 88.0%
The specificity for DecisionTreeClassifier is 96.02%
```



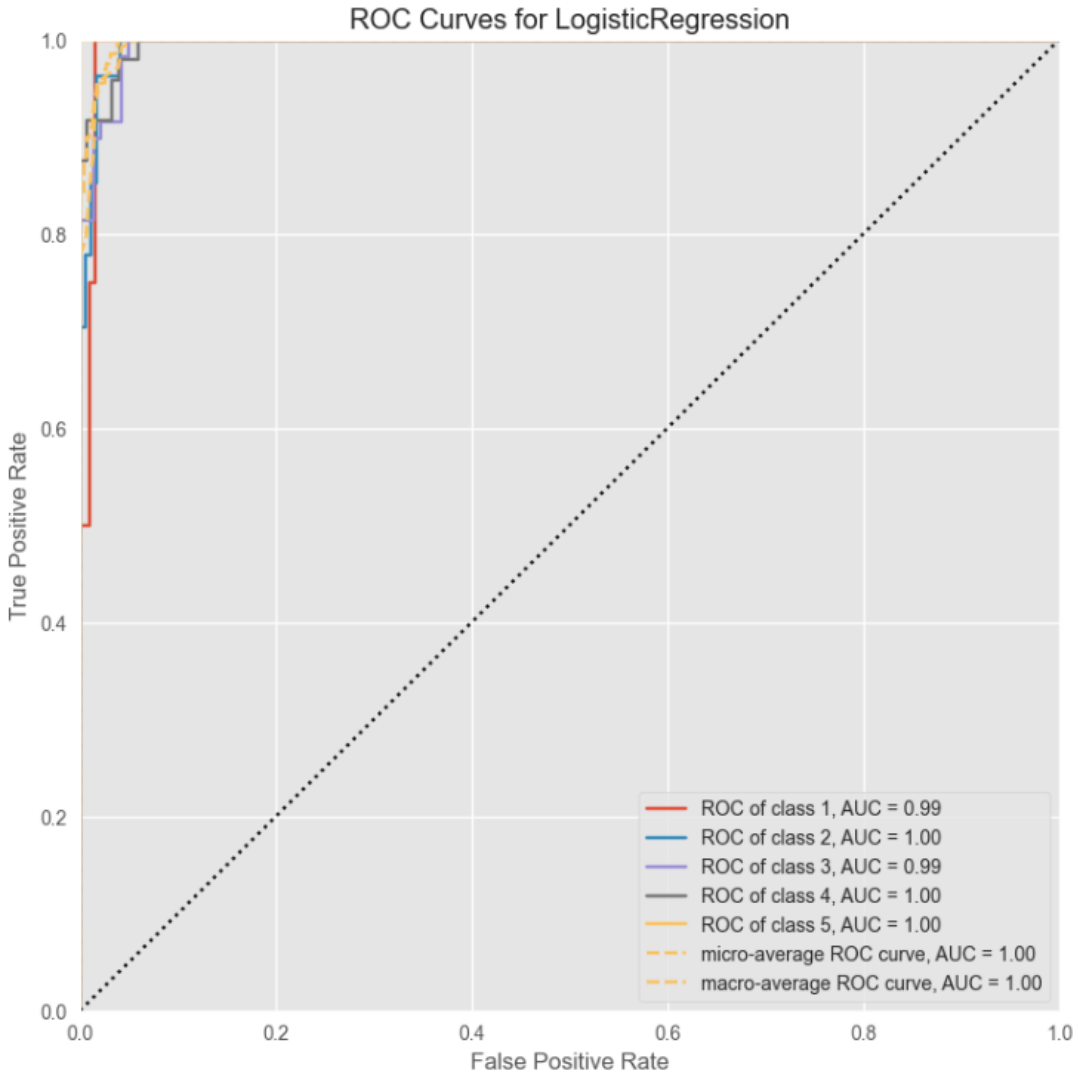
- Evaluation results for K-neighbors classifier

```
------------------------------------ KNeighborsClassifier ------------------------------------

The confusion matrix for KNeighborsClassifier:
[[ 2  2  0  0  0]
 [ 4 12 11  0  0]
 [ 1  7 39 12  0]
 [ 0  0 19 24  5]
 [ 0  0  0 13 49]]
The accuracy score for KNeighborsClassifier is 63.0%
The balanced accuracy score for KNeighborsClassifier is 57.92%
The sensitivity for KNeighborsClassifier is 63.0%
The specificity for KNeighborsClassifier is 87.9%
```



- Evaluation results for Logistic regression classifier

```
---------------------------------------- LogisticRegression ----------------------------------------

The confusion matrix for LogisticRegression:
[[ 2  2  0  0  0]
 [ 0 26  1  0  0]
 [ 0  1 54  4  0]
 [ 0  0  3 45  0]
 [ 0  0  0  1 61]]
The accuracy score for LogisticRegression is 94.0%
The balanced accuracy score for LogisticRegression is 85.99%
The sensitivity for LogisticRegression is 94.0%
The specificity for LogisticRegression is 98.14%
```



Conclusion :

From the above ROC curve we can conclude that **Logistic Regression** gives the best results for the classification as it has the largest AUC.