FACULTY OF ENGINEERING
EGYPT UNIVERSITY OF INFORMATICS

Exam Score Prediction Using Big Data Analytics and NoSQL Databases

CSE471 – Big Data Analytics projects

Nadeen Nadir 21-101167
Doha Hafez 21-101136
Zeyad Ayman 21-101144

1. Description of Approach

1.1 Dataset Description

- Dataset name: **Exam_Score_Prediction**
- Description:
  The dataset contains information about student's age , gender ,course, study behavior, class attendance , internet access and exam conditions.
- Number of features: 12
- Target variable: exam_score
- Task type: **Regression** (predicting exam score)

1.2 Data Storage and Big Data Component (MongoDB)

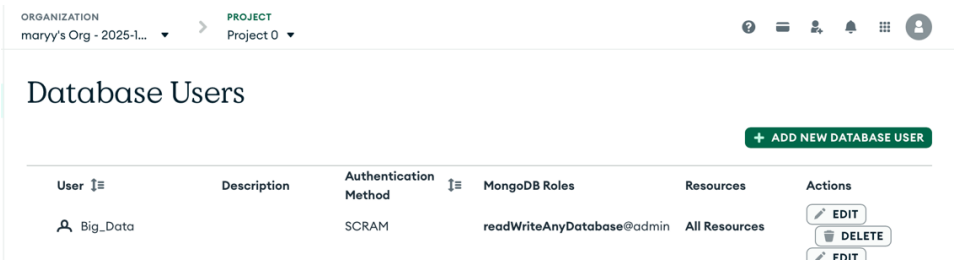
In this project, **MongoDB Atlas** was used as the Big Data component to store and manage the dataset. MongoDB is a NoSQL, document-based database that is suitable for handling large and semi-structured data with flexible schema design.

The dataset was uploaded to MongoDB Atlas as documents, where each document represents one student record. The database was then connected to a **Google Colab notebook** using the **PyMongo** library. All data retrieval operations for preprocessing, exploratory data analysis, and model preparation were performed directly from MongoDB.

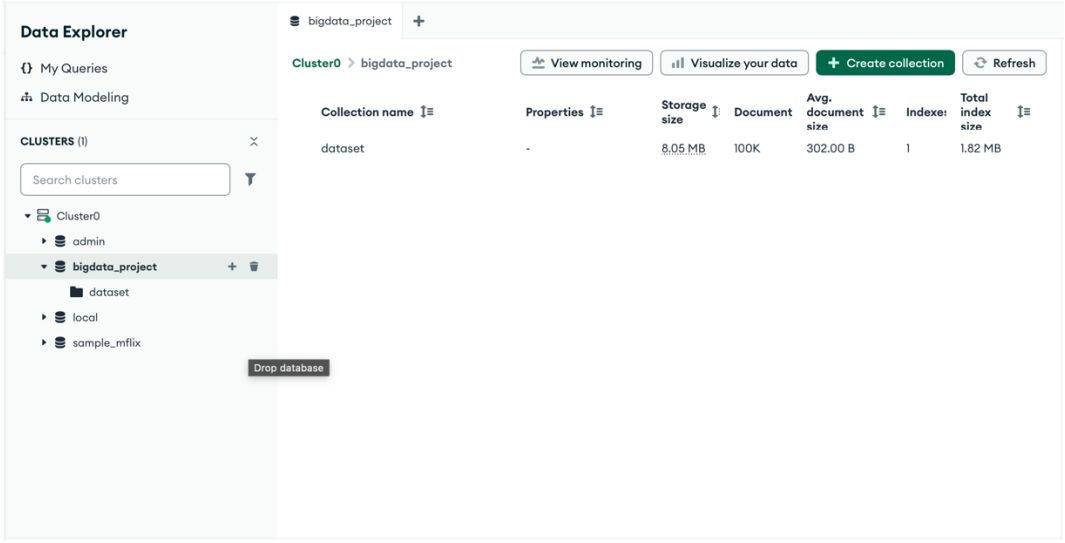
Using MongoDB allowed efficient querying, filtering,AND/OR conditions, and aggregation of the data before applying machine learning techniques.

Screenshots:

- MongoDB Atlas collection showing stored document :



[figure1: screenshot from mongo atlas]



[figure2: screenshot from mongo atlas- data set of the project]

- Python code showing successful connection to MongoDB:

```
#install mongo :
!pip install pymongo
```

```
Requirement already satisfied: pymongo in /usr/local/lib/python3.12/dist-packages (4.15.5)
Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in /usr/local/lib/python3.12/dist-packages (from pymongo) (2.8.0)
```

```
#making connection between mongo and collab note book
from pymongo import MongoClient

uri = "mongodb+srv://Big_Data:Thisisthenewpass@cluster0.jkadecy.mongodb.net/?appName=Cluster0"

client = MongoClient(uri)

db = client["bigdata_project"]
collection = db["dataset"]

print("Connected to MongoDB Atlas successfully!")
```

```
Connected to MongoDB Atlas successfully!
```

```
collection.insert_many(df.to_dict("records"))
```

[figure3: screenshot from notebook to connect it with mongo atlas]

1.3 Data Retrieval and NoSQL Queries

Several MongoDB queries were applied to explore and analyze the dataset:

- **Basic retrieval:** Loading records from MongoDB into the notebook using find()
- **Filtering:** Selecting students based on conditions such as exam score or study hours
- **Aggregation:** Computing statistics such as average exam score and grouping by categorical attributes (e.g., study method or internet access)

- **AND/OR :** Logical operators (AND) and (OR) were used to perform advanced filtering by combining multiple conditions in a single query.
- **Boolean / categorical**

These queries helped in understanding relationships between different factors and student performance before building the regression model.

Screenshots:

```
#filtering ( show students who get more than 80 )
list(collection.find({"exam_score": {"$gt": 80}}).limit(5))
```

[figure4: filtering query]

```
#filtering(show students who study more than 5 hours )
list(collection.find({"study_hours": {"$gt": 5}}).limit(5))
```

[figure5: filtering query]

```
#select specific columns:
list(collection.find(
    {},
    {"student_id": 1, "study_hours": 1, "exam_score": 1, "_id": 0}
).limit(5))
```

[figure6: filtering query]

```
#Aggregation:
list(collection.aggregate([
    {"$group": {
        "_id": None,
        "avg_score": {"$avg": "$exam_score"}
    }}
]))
```

```
[{'_id': None, 'avg_score': 62.513225}]
```

```
#Aggregation ( relation of study hours with grades ):
list(collection.aggregate([
    {"$group": {
        "_id": None,
        "avg_study_hours": {"$avg": "$study_hours"},
        "avg_exam_score": {"$avg": "$exam_score"}
    }}
]))
```

[figure7: Aggregation queries]

```
#find students with internet access

list(collection.find({"internet_access": "yes"}).limit(5))
```

[figure8: Boolean / Categorical Queries]

```
#find the students with no access to internet
list(collection.find({"internet_access": "no"}).limit(5))
```

[figure9: Boolean / Categorical Queries]

```
#and condition ( find students with good sleep quality & study hours more than 5 )
list(collection.find({
    "$and": [
        {"sleep_quality": "good"},
        {"study_hours": {"$gt": 5}}
    ]
}).limit(5))
```

[figure10: Multiple Conditions (AND / OR)]

```
#or conition (find students with high exam score greater than 85 & study hours more than 8)
list(collection.find({
    "$or": [
        {"exam_score": {"$gt": 85}},
        {"study_hours": {"$gt": 8}}
    ]
}).limit(5))
```

[figure11: Multiple Conditions (AND / OR)]

```
#show first 5 records
list(collection.find({}).limit(5))
```

[figure12: show the first 5 records]

1.4 Exploratory Data Analysis (EDA)

1.4.1 Summary statistics

We started by inspecting the dataset to get a basic understanding:

- **Dataset shape:** 6,000 records (rows) and 13 features (columns).
- **Features:** demographic, academic, and lifestyle variables such as age, gender, course, study hours, class attendance, sleep hours, sleep quality, study method, facility rating, exam difficulty, and the target variable `exam_score`.
- **Missing values:** None found.
- **Duplicate rows:** None detected.
- **Data types:** 7 numerical features and 6 categorical features.

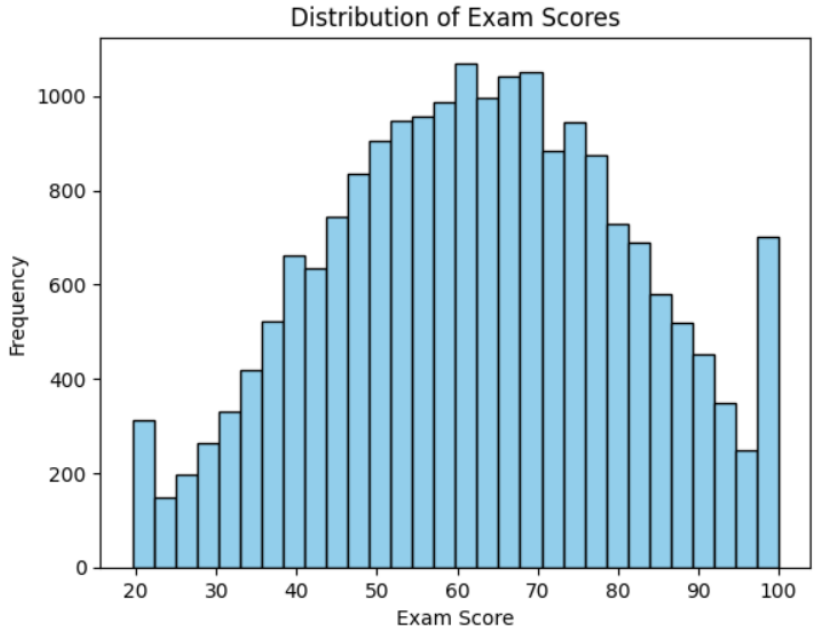
1.4.2 Data distributions

We explored the distribution of both numerical and categorical features:

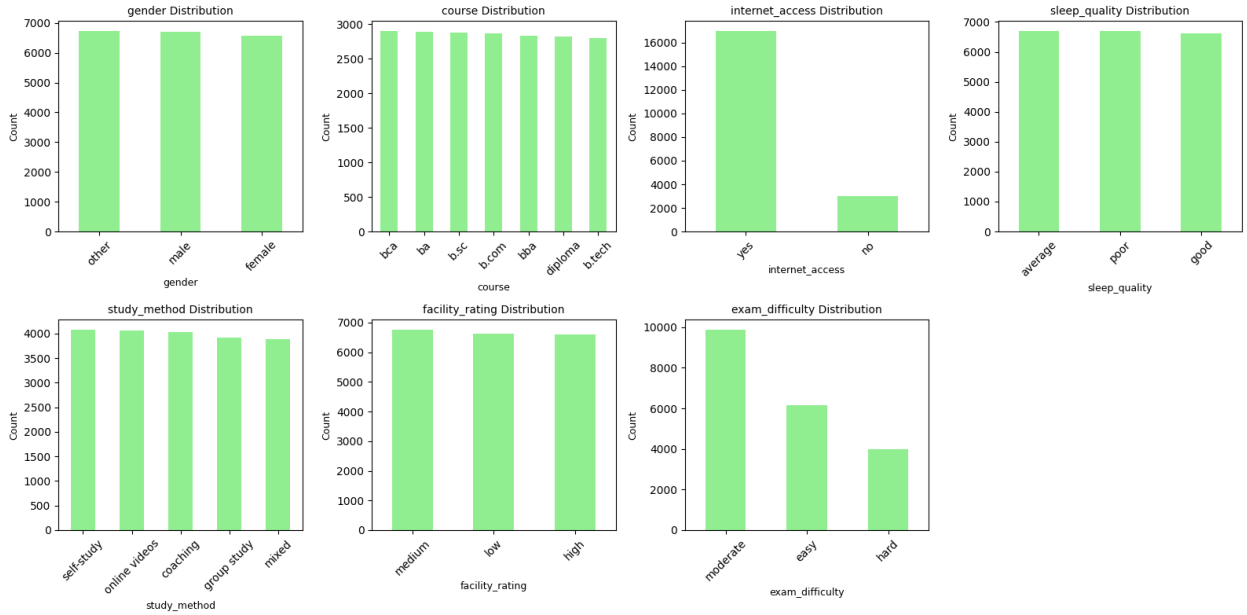
- **Numerical features:** We plotted a histogram of `exam_score` to check its distribution.
- **Categorical features:** Bar plots were created for each categorical variable (`gender`, `course`, `internet_access`, `sleep_quality`, `study_method`, `facility_rating`, `exam_difficulty`) to visualize the frequency of each category.

1.4.3 Visualizations

- **Distribution of Exam Scores**
  - Histogram shows most students scored between ~50 and 90.
  - A few students scored very low, indicating potential outliers.

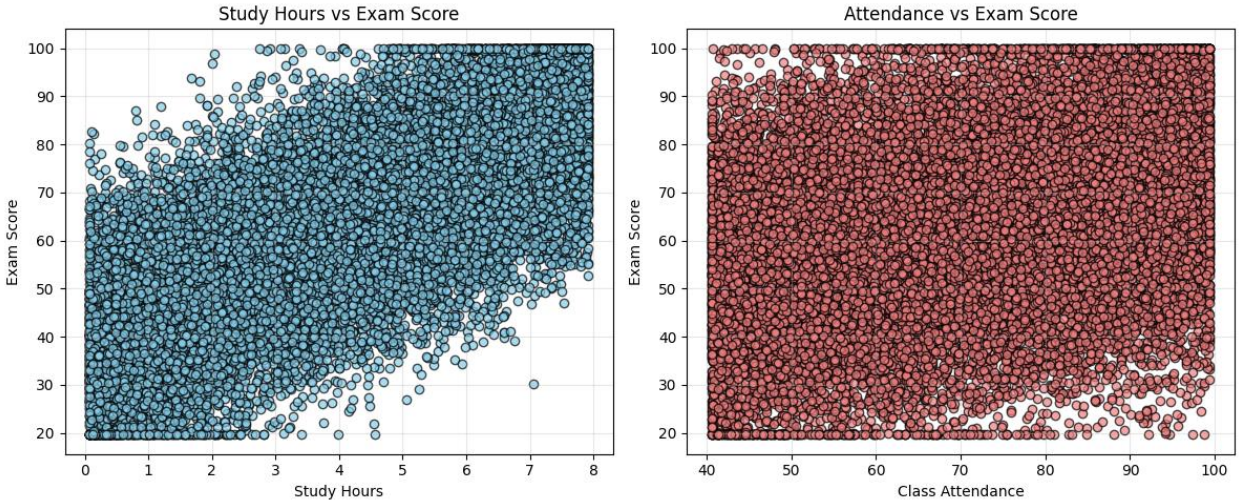


- **Categorical Feature Distributions**
  Bar plots revealed that some categories are more frequent than others, e.g., certain courses and study methods have more students.

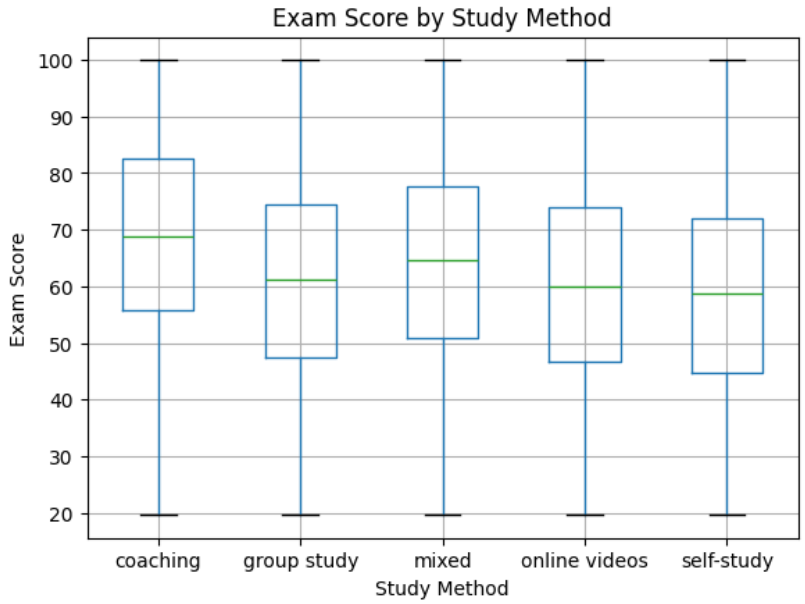


- **Scatter Plots**
  - **Study Hours vs Exam Score:** Positive trend; students who studied more tend to score higher.

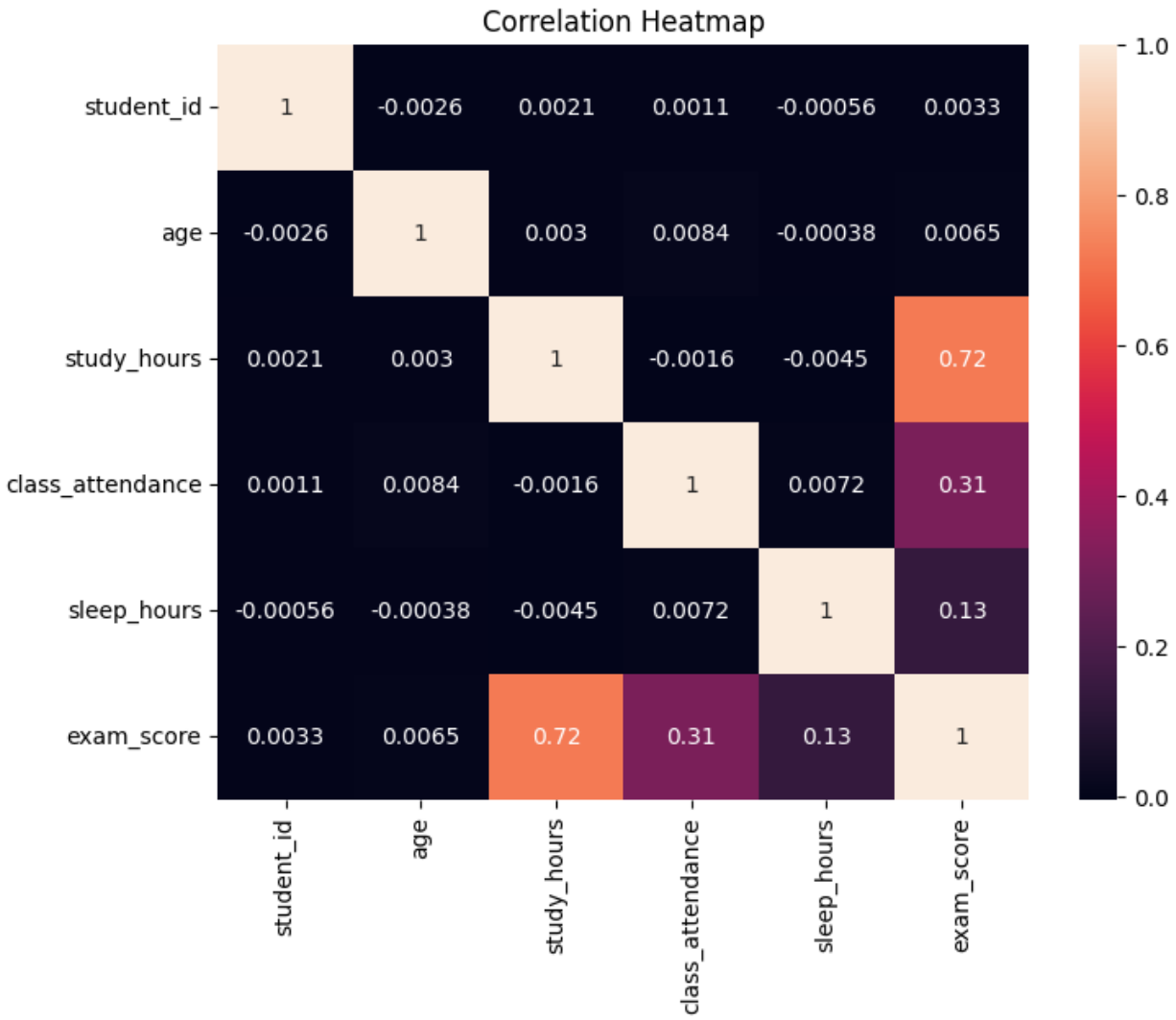
– **Class Attendance vs Exam Score:** Positive trend; higher attendance generally corresponds to higher scores.



- **Box Plot of Exam Score by Study Method**
  – Students using different study methods showed variation in exam scores.
  – Group study and coaching methods seem to have slightly higher median scores compared to self-study and online videos.



- **Correlation Heatmap**
  – A heatmap of numerical features showed that `study_hours` and `class_attendance` have moderate positive correlation with `exam_score`.
  – Other numerical features have weak or no correlation with the target.



1.4.4 Key observations

- No missing or duplicate data, which simplifies preprocessing.
- Exam scores are moderately distributed with most students scoring in the mid-range.
- Study hours and attendance are positively associated with exam performance.
- Categorical variables (e.g., study method, course) show some imbalance in frequencies, which could impact model performance if not handled carefully.
- The dataset appears well-structured and suitable for predictive modeling.

1.5 Machine Learning Model (Regression)

1.5.1 Selected Model

For this project, a Linear Regression model was selected to predict students' exam scores. Linear Regression is a supervised learning algorithm that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. This model was chosen due to its simplicity, interpretability, and effectiveness for continuous numerical prediction tasks such as exam score estimation.

1.5.2 Selected Feature

The target variable for the regression task is `exam_score`. All remaining dataset attributes were used as input features, including both numerical and categorical variables. Numerical features (such as study hours and class attendance) were directly included, while categorical features (such as study method) were transformed using **one-hot encoding** to convert them into a numerical format suitable for the regression model. This approach ensures that all relevant factors contributing to exam performance are incorporated into the model.

1.5.3 Train/Test Split

The dataset was divided into training and testing sets using an **80/20 split**. Specifically, 80% of the data was used to train the model, while the remaining 20% was reserved for evaluating its performance on unseen data. This split helps assess the model's generalization capability and prevents overfitting. The split was performed randomly with a fixed random seed to ensure reproducibility.

1.5.4 Model Training Process

Before training, feature scaling was applied using standardization, ensuring that all features have a mean of zero and a standard deviation of one. This step is important for linear regression, as it improves numerical stability and ensures that features with larger scales do not dominate the learning process.

The Linear Regression model was then trained using the scaled training data. During training, the model learned the optimal coefficients that minimize the difference between the predicted and actual exam scores by reducing the mean squared error. Once trained, the model was used to generate predictions on the test set for evaluation using standard regression metrics.

2. Output and Results

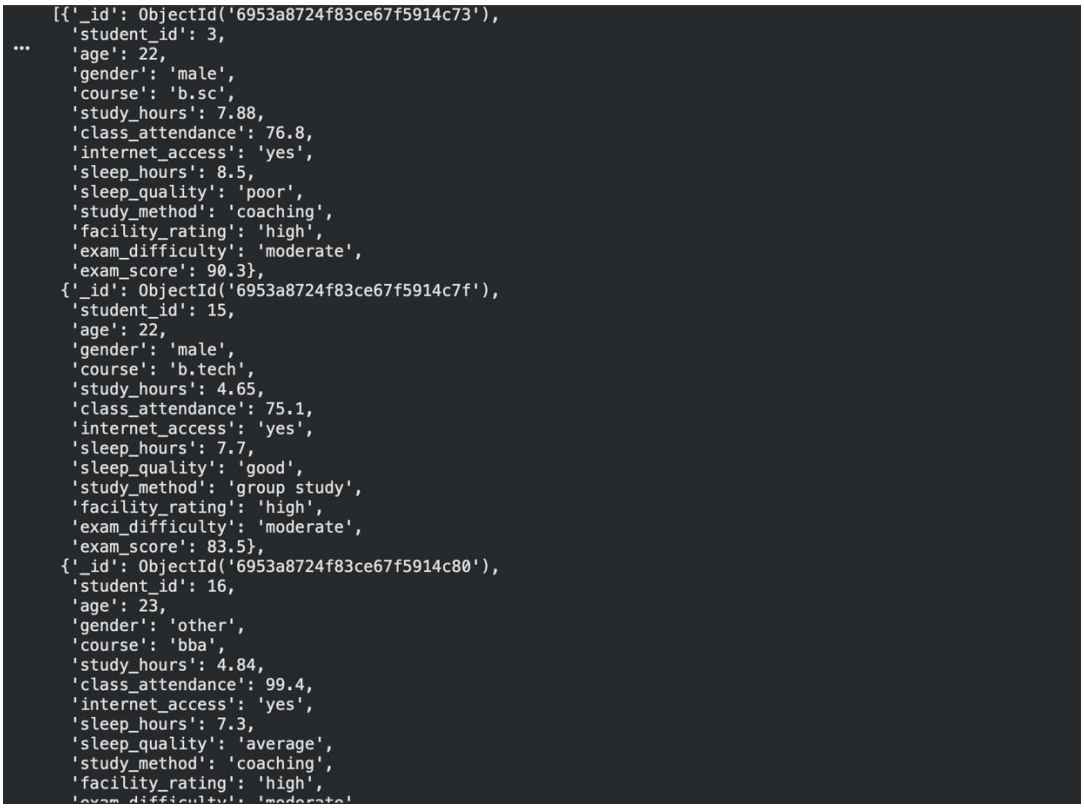
2.1 MongoDB Query Results

MongoDB queries successfully retrieved meaningful insights from the dataset. Filtering queries identified students with low or high performance, while aggregation queries showed how factors such as study method and sleep quality affect exam scores.

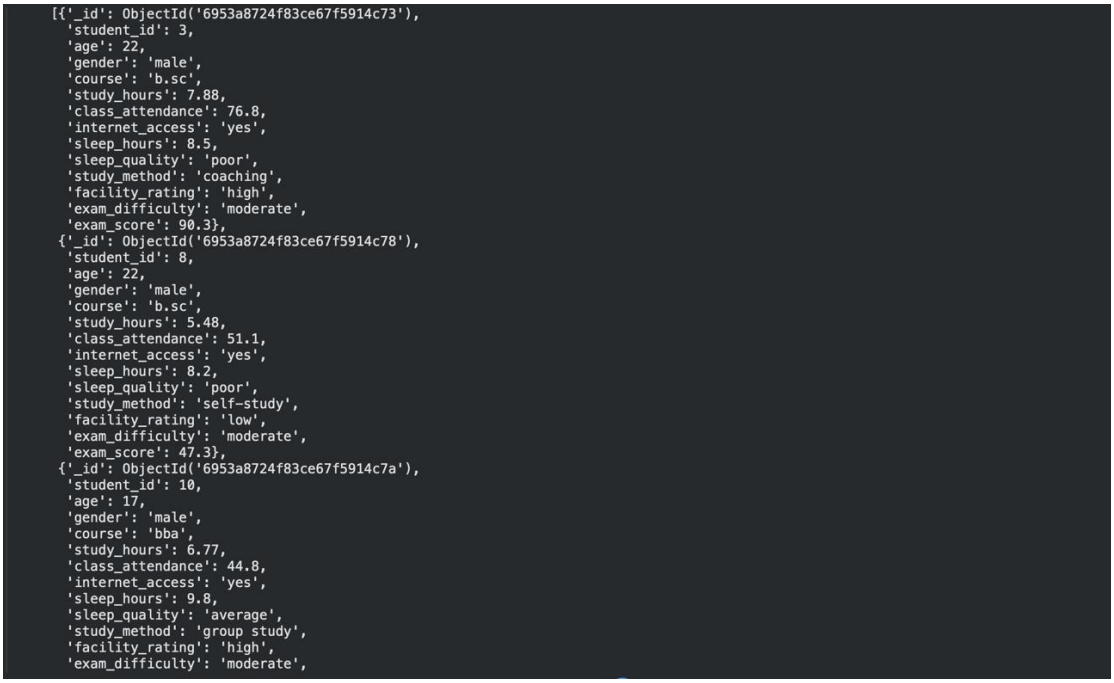
These results demonstrate the effectiveness of using MongoDB as a NoSQL database for data exploration in a Big Data analytics pipeline.

- Screenshot:

[figure 12: show the first 5 records – note : due to screen shot limit screen it just show 3 here]



[figure 12: show the records with exam score higher than 80 – note : due to screen shot limit screen it just show 3 here]



[figure 12: show the records study hours more than 5 – note : due to screen shot limit screen it just show 3 here]

```
[{'student_id': 1, 'study_hours': 2.78, 'exam_score': 58.9},
 {'student_id': 2, 'study_hours': 3.37, 'exam_score': 54.8},
 {'student_id': 3, 'study_hours': 7.88, 'exam_score': 90.3},
 {'student_id': 4, 'study_hours': 0.67, 'exam_score': 29.7},
 {'student_id': 5, 'study_hours': 0.89, 'exam_score': 43.7}]
```

[figure 12: select specific columns]

```
#number of students who got grater than or equal 50 :
collection.count_documents({"exam_score": {"$gte": 50}})
```

```
72980
```

[figure 12: show number of students with score grater than or equal 50]

```
#Aggregation:
list(collection.aggregate([
    {"$group": {
        "_id": None,
        "avg_score": {"$avg": "$exam_score"}
    }}
]))
```

```
[{'_id': None, 'avg_score': 62.513225}]
```

[figure 12: count the average of exam score]

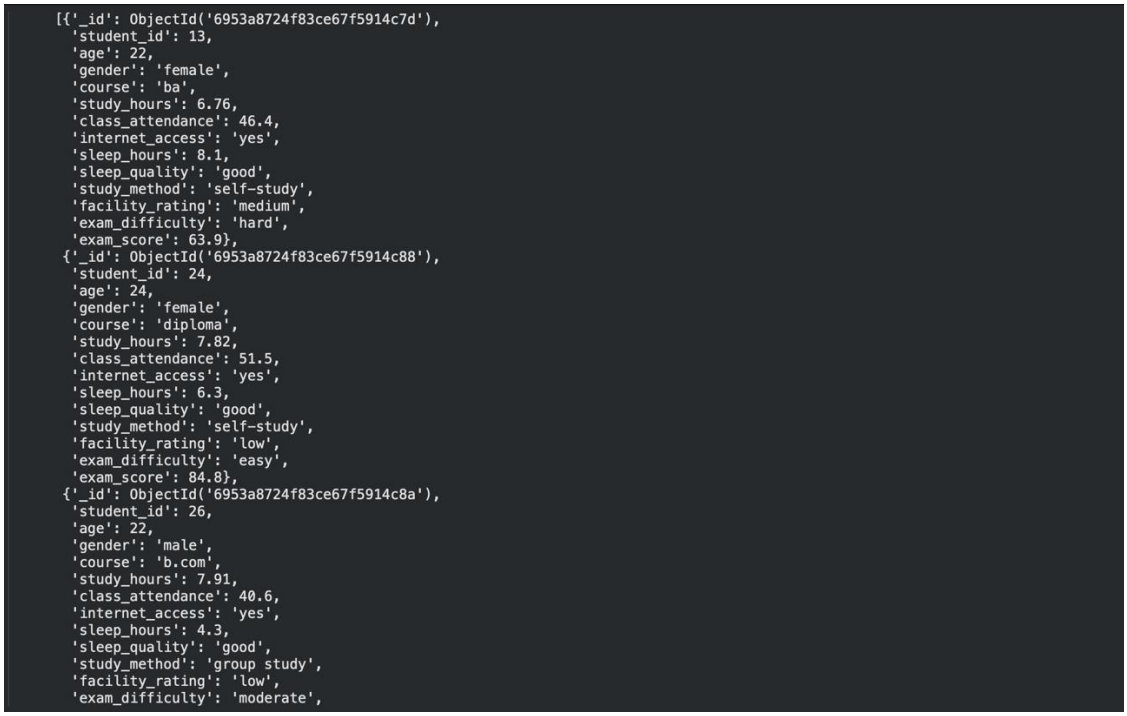
```
#Aggregation ( relation of study hours with grades ):
list(collection.aggregate([
    {"$group": {
        "_id": None,
        "avg_study_hours": {"$avg": "$study_hours"},
        "avg_exam_score": {"$avg": "$exam_score"}
    }}
]))
```

```
[{'_id': None, 'avg_study_hours': 4.0076035, 'avg_exam_score': 62.513225}]
```

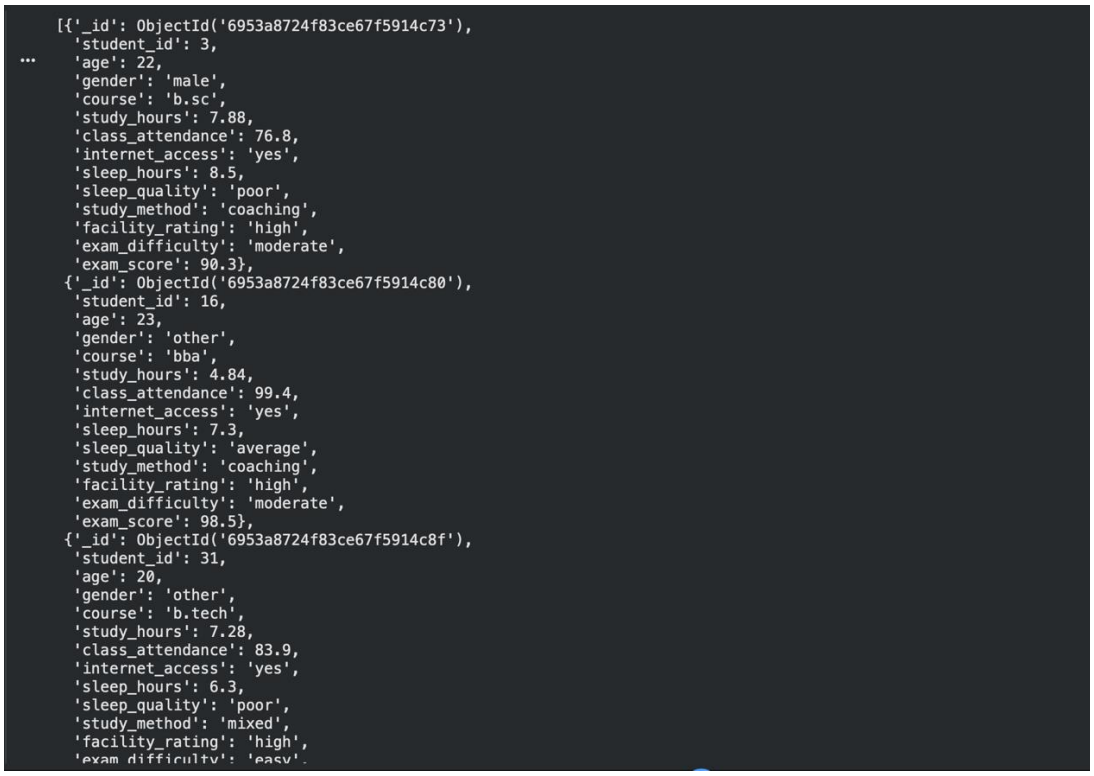
[figure 12: show the average of study hours and avg of exam score]

[figure 12: show the students with access to internet (yes) – note : due to screen shot limit screen it just show 3 here]

[figure 12: show students with no access to internet– note : due to screen shot limit screen it just show 3 here]



[figure 12: show students with “good” sleep quality AND study hours more than 5 – note : due to screen shot limit screen it just show 3 here]



[figure 12: show students with exam score grater than “85” OR study hours more than 8]

2.2 Regression Model Results

2.2.1 Model performance metrics

The performance of the Linear Regression model was evaluated using standard regression metrics, including **Root Mean Squared Error (RMSE)** and the **coefficient of determination (R^2)**. RMSE measures the average magnitude of prediction errors in the same units as the target variable, while R^2 indicates how well the model explains the variance in exam scores.

The obtained results demonstrate that the model achieves a relatively low RMSE, indicating accurate predictions with minimal deviation from actual exam scores. Additionally, the R^2 value shows that a substantial portion of the variability in exam performance is explained by the selected features, suggesting that the model fits the data reasonably well.

2.2.2 Predicted vs actual values

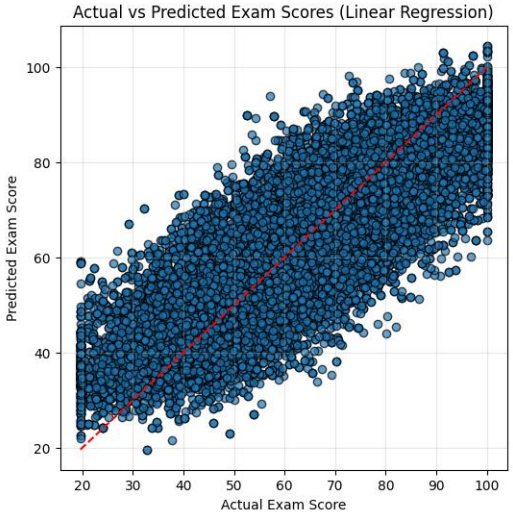
To visually assess model performance, a scatter plot comparing **predicted exam scores** against **actual exam scores** was generated. Most data points lie close to the diagonal reference line, which represents perfect prediction. This alignment indicates that the model’s predictions closely match the true exam scores across the dataset, with only minor deviations for some samples.

2.2.3 Interpretation of results

The regression results suggest that factors such as study habits and attendance have a measurable linear relationship with exam performance. The model effectively captures these relationships and provides reliable predictions within the observed data range. However, some prediction errors remain, likely due to unmodeled factors such as individual learning differences or external influences not included in the dataset.

Overall, the Linear Regression model demonstrates satisfactory performance for exam score prediction while maintaining simplicity and interpretability, making it suitable for this application.

Screenshot:



[figure 13: Predicted vs Actual Exam Scores using Linear Regression]

3. Conclusion

This project applied Big Data Analytics techniques to predict student exam performance. MongoDB was used as a NoSQL database to efficiently store and retrieve the dataset, while regression analysis was used to model the relationship between study behavior and exam scores. The results highlight the importance of data-driven analysis in educational performance prediction.

4. Code

Github Repository Link