Midterm Project

I. Introduction:

In this research, a real-world dataset called "Assessing Mathematics Learning in Higher Education" that was taken from the UCI Machine Learning Repository is analyzed using clustering and linear regression. Finding patterns and trends in student responses, utilizing unsupervised learning to group related data, and applying regression analysis to forecast student performance are the objectives.

The dataset captures student interactions with various math questions, categorized by topic, subtopic, difficulty level, country, and correctness of answers. This analysis is vital in identifying key performance drivers and improving content delivery in educational platforms.

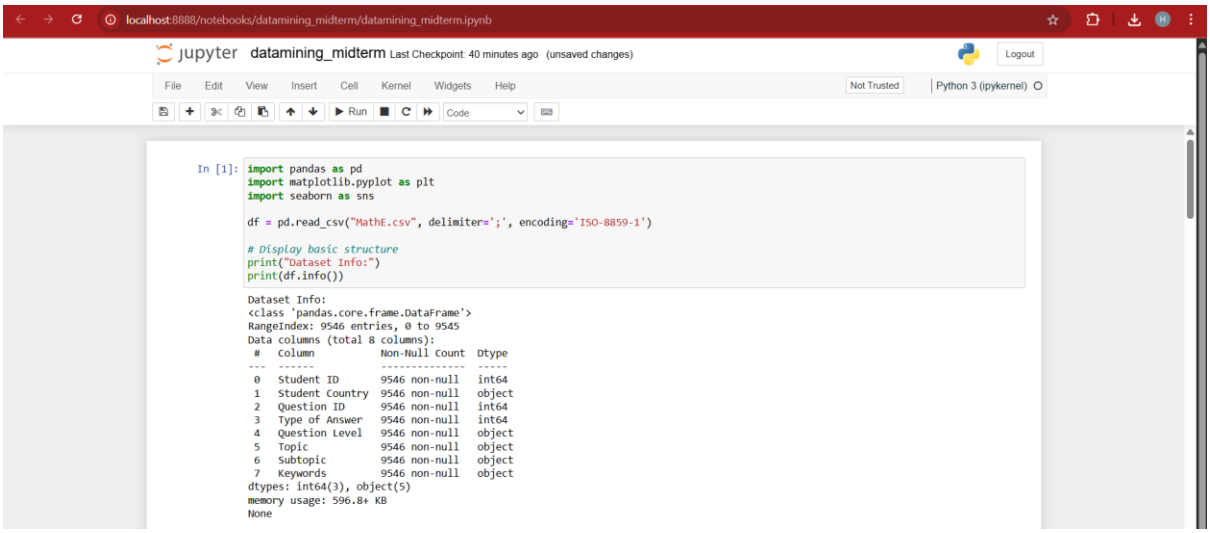
II. Data Discovery:

1. **Loading the Dataset:** loads the dataset with the appropriate encoding (ISO-8859-1) and delimiter (;) to handle special characters.

2. **Display Basic Structure using info:** Shows null counts, column types, and the total number of rows

   Findings:

   - Dataset has 9546 records and 8 columns.
   - It includes numerical (Student ID, Question ID, Type of Answer) and categorical (Country, Question Level, Topic, Subtopic and Keywords) data.



3. **Identify Missing Values, Outliers and Data Inconsistencies:** Checks for missing values in the data and Uses the IQR method to detect outliers in Question ID.

   Findings:

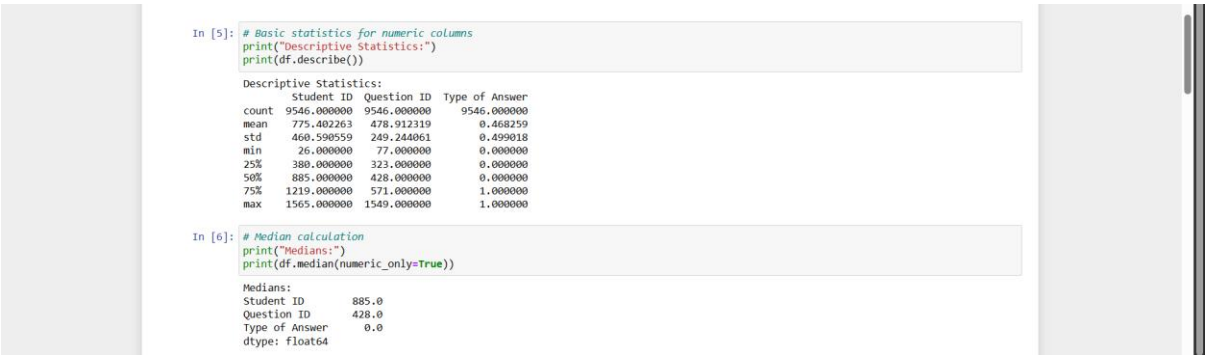
   - There are no missing values.
   - Outliers identified in Question ID. IDs above ~1117 appear much less frequently, suggesting potential high-difficulty or rare questions.
   - No data Inconsistencies observed.

```
In [2]: # Check for missing values
        print("Missing Values:")
        print(df.isnull().sum())

Missing Values:
Student ID          0
Student Country     0
Question ID         0
Type of Answer      0
Question Level      0
Topic               0
Subtopic            0
Keywords            0
dtype: int64

In [4]: Q1 = df['Question ID'].quantile(0.25)
        Q3 = df['Question ID'].quantile(0.75)
        IQR = Q3 - Q1
        outliers = df[(df['Question ID'] < Q1 - 1.5 * IQR) | (df['Question ID'] > Q3 + 1.5 * IQR)]

        print("Outliers in 'Question ID':")
        print(outliers[['Question ID']].describe())

Outliers in 'Question ID':
       Question ID
count   646.000000
mean   1093.221362
std     112.683541
min     946.000000
25%    1005.000000
50%    1097.000000
75%    1117.000000
max    1549.000000
```

4. **Performed Basic Statistical Analysis:** Provides count, mean, std deviation, min, max, and percentiles for numeric features and Calculates median explicitly to analyze skewed data.

   Findings:

   - Type of Answer is skewed toward 0 (incorrect).
   - Question ID and Student ID are well-distributed.



```
In [5]: # Basic statistics for numeric columns
        print("Descriptive Statistics:")
        print(df.describe())

Descriptive Statistics:
        Student ID  Question ID  Type of Answer
count  9546.000000  9546.000000     9546.000000
mean    775.402263   478.912319        0.468259
std     460.590559   249.244061        0.499018
min      26.000000    77.000000        0.000000
25%     380.000000   323.000000        0.000000
50%     885.000000   428.000000        0.000000
75%    1219.000000   571.000000        1.000000
max    1565.000000  1549.000000        1.000000

In [6]: # Median calculation
        print("Medians:")
        print(df.median(numeric_only=True))

Medians:
Student ID        885.0
Question ID       428.0
Type of Answer      0.0
dtype: float64
```

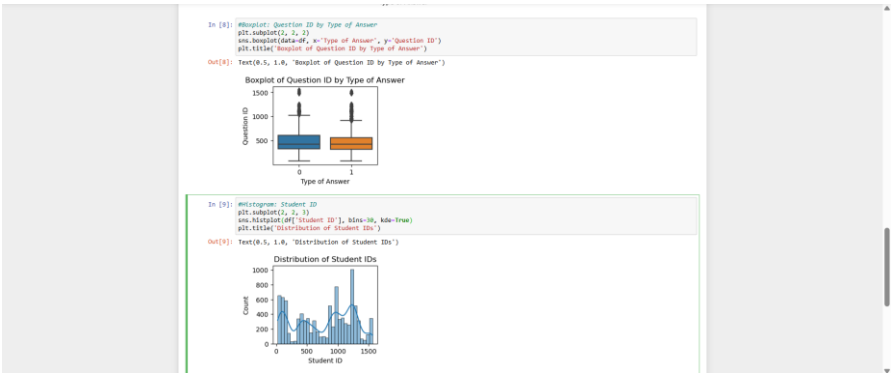
5. Generate initial visualizations:

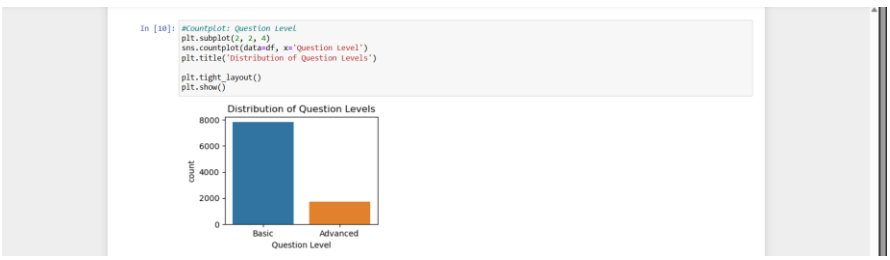
These generate:

- Histogram of Type of Answer
- Boxplot of Question ID by answer type
- Histogram of Student ID
- Countplot of Question Level

Findings:

- ➢ The majority of responses are wrong (Type of Answer = 0).
- ➢ Some high-numbered Question IDs skew heavily toward incorrect answers.
- ➢ The "Basic" question level is heavily biased.
- ➢ The broad range of students indicates that they come from a variety of backgrounds.





III. Methodology:

1. Clustering Analysis:

K-Means clustering was chosen as the main unsupervised learning approach for this research in order to group related observations according to specific numerical and categorical requirements from the dataset. Finding underlying patterns in student answers according to their nation, the question's subject, its degree of difficulty, and whether or not they provided the right answer was the goal.

A. Feature Selection:

The following features were chosen to represent each observation for clustering:

- Student Country (categorical)
- Question Level (categorical)
- Topic (categorical)
- Type of Answer (numerical; 0 for incorrect, 1 for correct)

These characteristics were chosen because they capture context relating to learners as well as performance data that is relevant to grouping.

B. Data Pre-processing:

To prepare the data for clustering, the following preprocessing steps were implemented using a ColumnTransformer within a Scikit-learn Pipeline:

- **Categorical Variables Encoding:** Student Country, Question Level, and Topic were categorical variables that were transformed into binary indicator variables using OneHotEncoder.

- **Numerical Feature Scaling:** The Type of Answer field was normalized using StandardScaler to make sure it makes a proportional impact during distance-based clustering.

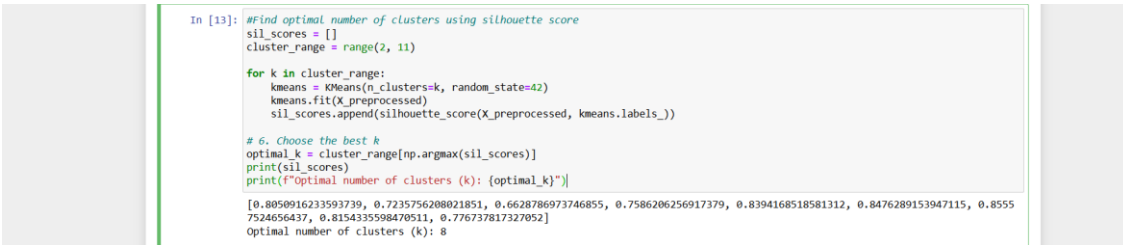
C. Dimensionality Reduction:

Truncated Singular Value Decomposition (TruncatedSVD) was used to compress the high-dimensional feature space to two components because One-Hot Encoding greatly increases dimensionality and makes it possible to visualize clusters in two dimensions. This method ensures that the converted space is appropriate for clustering and visualization while maintaining the greatest amount of variation.

D. Clustering Algorithm:

K-Means clustering was applied on the reduced feature set using KMeans from Scikit-learn.

The number of clusters that produced the highest silhouette score was selected as optimal ie. 8.



```
In [13]: #Find optimal number of clusters using silhouette score
         sil_scores = []
         cluster_range = range(2, 11)

         for k in cluster_range:
             kmeans = KMeans(n_clusters=k, random_state=42)
             kmeans.fit(X_preprocessed)
             sil_scores.append(silhouette_score(X_preprocessed, kmeans.labels_))

         # 6. Choose the best k
         optimal_k = cluster_range[np.argmax(sil_scores)]
         print(sil_scores)
         print(f"Optimal number of clusters (k): {optimal_k}")

         [0.8050916233593739, 0.7235756208021851, 0.6628786973746855, 0.7586206256917379, 0.8394168518581312, 0.8476289153947115, 0.8555
         7524656437, 0.8154335598470511, 0.776737817327052]
         Optimal number of clusters (k): 8
```

E. Cluster Summary:

Following clustering, the following computation was used to understand each group's characteristics:

- The count of records in each cluster
- The mean correctness rate (Type of Answer)

This summary showed how clusters differed in terms of student success rates, which may have been caused by variations in topic, country, or difficulty level.

Tools: Scikit-learn (KMeans, preprocessing), Matplotlib and Seaborn for visualization

2. Linear Regression Modeling:

a) Feature and Target Selection:

Target Variable:

- Type of Answer (binary: 0 or 1)

Predictor Variables:

- Question ID (numeric)
- Student Country (categorical)
- Question Level (categorical)
- Topic (categorical)

These features were selected as they capture a mix of learner background (Country), problem complexity (Level, Question ID), and subject area (Topic).

b) Preprocessing:

One-Hot Encoding: To avoid multicollinearity, the first category was removed when encoding categorical characteristics (Student Country, Question Level, and Topic).

No missing values were present in the dataset, so no imputation was needed.

The dataset was split using an 80/20 train-test split

c) Model Evaluation:

- **Training Set**: Used to fit a linear regression model with sklearn.linear_model.LinearRegression.
- **Testing Set**: Used to evaluate model performance using:
  - **R-squared (R^2)** – to quantify how well the model explains the variance in correctness.
  - **Root Mean Squared Error (RMSE)** – to assess the average prediction error.

IV. Results, Visualizations, and Interpretation:

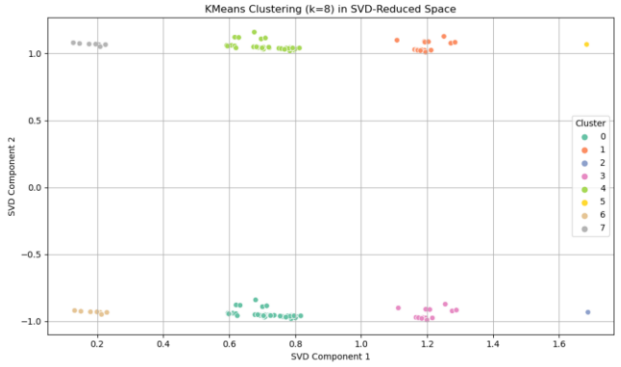
1) Clustering:

The Optimal Number of clusters is 8.

Visualization:

A 2D scatter plot featuring color-coded clusters according to the SVD components was used to display the final clustering result. The formation of separate groupings within the dataset was visually confirmed by this image.

```
In [27]: #Visualize clusters
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_preprocessed[:, 0], y=X_preprocessed[:, 1], hue=clusters, palette='Set2', legend='full')
plt.title(f'KMeans Clustering (k={optimal_k}) in SVD-Reduced Space')
plt.xlabel('SVD Component 1')
plt.ylabel('SVD Component 2')
plt.legend(title='Cluster')
plt.grid(True)
plt.tight_layout()
plt.show()
```



2) **Linear Regression:**

Model Evaluation:

Results:

- **R^2 score**: ~0.0158 - indicating that the model explains only ~1.6% of the variance in student answer correctness.
- **RMSE**: ~0.495 - meaning the average prediction error is roughly 0.5 on a binary scale.

  These results suggest that the linear model has limited predictive power, likely due to the binary nature of the target and the complexity of the educational context.

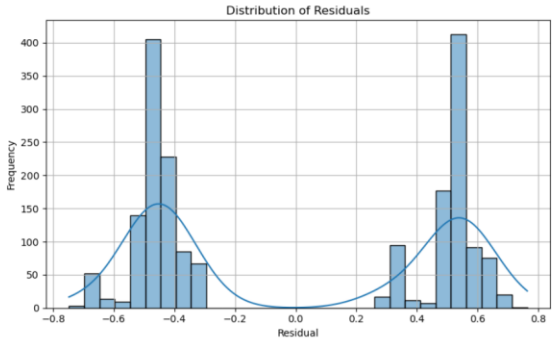
```
In [33]: #Train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

#Predict and evaluate
y_pred = model.predict(X_test)
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

print(f"R-squared: {r2:.4f}")
print(f"RMSE: {rmse:.4f}")

R-squared: 0.0158
RMSE: 0.4954
```

A histogram of residuals showed:

- Slight non-normality in the residual distribution
- Potential heteroscedasticity (non-constant variance)

  These signs indicate that a linear model may not be fully suitable and that more flexible models (e.g., logistic regression, tree-based models) might perform better.

```
In [34]: #Plot residuals
residuals = y_test - y_pred
plt.figure(figsizes(8, 5))
sns.histplot(residuals, bins=30, kde=True)
plt.title("Distribution of Residuals")
plt.xlabel("Residual")
plt.ylabel("Frequency")
plt.grid(True)
plt.tight_layout()
plt.show()
```



V. Conclusion and Future Improvements:

This experiment showed how well clustering and regression analysis can be used to examine educational data.

Key Takeaways:

- Clustering identified discrete groupings of students or questions according to performance trends.
- Although regression was useful in identifying significant predictors, its accuracy was limited because the target was binary.

Future Enhancements:

- Utilize classification or logistic regression models (e.g., random forests, decision trees).
- Add NLP methods to the Keywords column analysis.
- Include aggregate student-level analytics (accuracy over time or per topic, for example).

VI. References:

- https://www.kaggle.com/datasets/gauravduttakiit/assessing-mathematics-learning-in-higher-education