# BrightPath Academy Project Agenda

**Supervised Learning Model**

**Why Supervised Learning?**  
In supervised learning, we know both the input features and the target output, which we aim to predict or classify. This approach allows us to train models to learn the relationship between input and output.

**Algorithms to Use:**

* Logistic Regression
* Random Forest
* XGBoost
* Deep Learning (e.g., neural networks)

**1. Define Problem Statement**

Clearly define the goal of the project—predicting the grade class of students based on various features like study time, tutoring, and parental support.

**2. Hypothesis Generation**

Generate hypotheses about potential relationships between input features (e.g., study time, tutoring) and the target output (grade class).

**3. Load Data into the System**

Load the data from a CSV file for analysis and model training.

**4. Understand the Data**

**Data Preparation and Preprocessing:**

* **Handle Missing Data:**
  + Impute missing values.
  + Drop rows or columns with excessive missing data.
* **Categorical Encoding:**
  + Use Label Encoding to convert categorical variables into numeric labels.
* **Feature Scaling:**
  + Apply Min-Max Scaling or Standardization to features to help models like Logistic Regression and XGBoost perform better.
* **Outlier Detection:**
  + Identify and handle outliers that may skew the model's results.

**5. Exploratory Data Analysis (EDA)**

* **Univariate Analysis:**
  + Examine the distribution of each feature using histograms, box plots, etc.
* **Bivariate Analysis:**
  + Check correlations between features and the target variable using scatter plots or correlation heatmaps.
* **Class Imbalance:**
  + Check for imbalanced classes (e.g., grade classes A, B, C, D, F). If needed, apply techniques like SMOTE (Synthetic Minority Over-sampling Technique) or use class weights during model training.

**6. Missing Value and Outlier Treatment**

Address missing values and outliers by applying appropriate treatments.

**7. Evaluation Metrics for Classification Problem (Post-Model Building)**

After training your models, evaluate their performance using the following metrics:

* **Accuracy:** The percentage of correct predictions.
* **Precision, Recall, F1-Score:** Useful for imbalanced datasets.
* **Confusion Matrix:** Visualize model performance for each class.

**8. Feature Engineering**

Enhance model performance by creating new features or transforming existing ones. For example:

* Combine related features (e.g., "StudyTimeWeekly" and "Tutoring" could form a "Study Support" feature).
* Create interaction features if needed (e.g., "ParentalSupport \* StudyTimeWeekly").

**9. Model Building**

* **Step 1: Split the Data:**
  + Split data into training (e.g., 70%) and testing sets (e.g., 30%) to evaluate performance on unseen data.
* **Step 2: Train Multiple Models:**
  + **Logistic Regression / Multiple Logistic Regression:** Start with logistic regression as a baseline model to understand how each feature impacts the target variable.
  + **Random Forest:** Suitable for handling multiple features and complex relationships. It also provides feature importance.
  + **XGBoost:** A powerful gradient boosting model ideal for large datasets with many features and internal handling of missing data.

**10.Model Building (Part 2):**

* **Deep Learning (Optional):** If other models do not capture relationships effectively, consider using a neural network. Neural networks handle non-linear relationships but require more data and computational resources.

**11. Model Deployment**

Deploy the trained model on [**https://render.com/**](https://render.com/), and create a dashboard to visualize and interact with the model's predictions.

Green-easy = 3 tasks

Orange -intermediate =4tasks

Red-Difficult= 3tasks  
  
**Each member gets one easy task, one intermediate task, and one hard task.**