MTL782: Assignment 1

Professor	Dr. Niladri Chatterjee
Assignment TAs	Atul, Kanav, Nancy, Shreya
Total Weightage	15
Submission Deadline	**11:59 pm, Sunday Feb 9, 2025**

Guidelines

- Do not plagiarize, as we will be running plagiarism checkers on your assignments.
- The code will be checked manually, and any mismatch will result in a penalty.
- Prepare a **LaTeX report** detailing all your analysis, observations, and plots.
- All codes will be run to verify the outputs. Faulty submissions will lead to **serious penalties**.
- Kaggle submissions done after the deadline is over will not be considered.
- Make sure to train the model with the same training data in both the non-competitive part and competitive part.

Standard ML Pipeline

1) Exploratory Data Analysis (EDA)

What is EDA? Exploratory Data Analysis (EDA) is an essential step in understanding the data, its patterns, anomalies, relationships, and structures. It involves both statistical and graphical techniques to summarize the data's main characteristics, often visualizing the underlying patterns that will inform further preprocessing and modeling choices.

Key Tasks in EDA

- **Data Overview**:
  - **Shape of the data**: Check the number of rows and columns.
  - **Missing values**: Identify columns with missing data and decide on strategies for handling them (e.g., imputation, removal).
- **Descriptive Statistics**:
  - Calculate summary statistics for numerical columns (mean, median, standard deviation, quartiles, etc.).
  - For categorical columns, analyze the frequency distribution of each category.
- **Data Distribution**:
  - Visualize the distribution of numerical features using histograms, density plots, and boxplots.
  - Visualize the distribution of categorical features using bar plots.
- **Correlation Analysis**:
  - For numerical features, compute correlation matrices to identify relationships between features.
  - Visualize correlations using heatmaps to understand linear dependencies.
- **Outlier Detection**:
  - Identify outliers using boxplots, scatter plots, or statistical methods (e.g., Z-score or IQR-based methods).

  - Decide how to treat outliers (e.g., removing, capping, or transforming them).

- **Feature Relationships**:
  - Explore relationships between features and the target variable using scatter plots (for continuous variables) or boxplots (for categorical variables).
  - Pairplot or scatter matrix can show interactions between multiple variables.

Link between EDA and Data Cleaning

- **Handling Missing Data**: Based on EDA, decide whether to fill missing values (imputation) or remove rows/columns with missing data.
- **Handling Duplicates**: Duplicates can be identified through EDA, and should be removed to prevent skewing model performance.
- **Outlier Removal**: Outliers detected during EDA can be removed or transformed during the data cleaning phase to prevent them from influencing model training.

Useful Links for EDA

- Essential Guide to Exploratory Data Analysis
- Comprehensive Guide for EDA
- Kaggle EDA Tutorial

2) Data Cleaning (Based on EDA Insights)

What is Data Cleaning? Data cleaning is a critical step in the preprocessing pipeline where we handle missing, incorrect, or inconsistent data identified during EDA. The goal is to prepare the data for modeling by resolving any issues that might interfere with the learning process.

Key Steps in Data Cleaning

- **Handling Missing Data**:
  - **Imputation**: Fill missing values using statistical methods (mean, median, mode) or predictive modeling techniques (e.g., k-NN imputation).
  - **Removal**: Drop rows or columns with too many missing values or where missing data is non-informative.
- **Removing Duplicates**:
  - Drop duplicate rows that do not provide new information.
- **Handling Outliers**:
  - If outliers are detected in EDA, they can be handled by transforming (e.g., using log transformation), removing, or capping them based on domain knowledge.

Useful Links for Data Cleaning

- Introduction to Handling Missing Values
- Complete Guide to Data Cleaning
- Data preprocessing for Machine Learning

3) Data Splitting (Train, Validation, Test)

What is Data Splitting? Data splitting ensures that your machine learning model is trained on a portion of the data and evaluated on a separate, unseen portion to prevent overfitting and assess real-world performance. This is crucial for ensuring that the model generalizes well to new data.

Key Concepts in Data Splitting

- **Training Set**: The largest subset, typically 70-80% of the data. The model is trained on this data.
- **Validation Set**: A smaller subset (10-20% of the data), used during model tuning and hyperparameter selection. It helps evaluate model performance during training and select the best model configuration.
- **Test Set**: A final subset, typically 10-20% of the data, used only for evaluating the final model. This dataset must remain untouched during model training to ensure an unbiased assessment of model performance.

Cross-Validation

- **K-Fold Cross-Validation**: The data is divided into k folds, and the model is trained k times, each time using a different fold as the validation set and the remaining folds for training.
- **Stratified Sampling**: Used especially in classification tasks to ensure that each split has a similar distribution of target classes as the original dataset.

Useful Links for Data Splitting

- Understanding Train Test Split
- Cross-Validation
- Splitting Data for Machine Learning Models

4) Data Types and Encoding Methods

Data refers to raw facts or values that are collected, processed, and analyzed to gain insights, make decisions, or perform tasks. In machine learning and analytics, data is typically categorized into two broad types: Categorical and Numerical.

1. Categorical Data

Categorical data represents values in distinct groups or categories. These are often non-numeric and describe qualities or characteristics.

- **Nominal Data**: Categories with no inherent order. Examples: Gender (Male, Female), Hair Color (Blonde, Brown, Black), Nationality (American, Indian, French)
- **Ordinal Data**: Categories with a meaningful order, but no numerical differences. Examples: Education Level (High School, Bachelor's, Master's), Customer Satisfaction (Poor, Fair, Good, Excellent), Rank in a Competition (1st, 2nd, 3rd)

2. Numerical Data

Numerical data represents measurable quantities and can be classified as discrete or continuous.

- **Discrete Data**: Countable, distinct values, often integers. Examples: Number of children, Number of cars in a parking lot, Age (in years)
- **Continuous Data**: Measurable values within a range, including fractions or decimals. Examples: Height (5.5 ft, 5.7 ft), Weight (65.4 kg), Temperature (22.5°C, 25.7°C)

Popular Methods to Process and Encode Data

- **Categorical Data**
  - **One-Hot Encoding**: Converts categories into binary columns. Best for nominal data (no order). Example: "Color" $\rightarrow$ [Red, Blue, Green] $\rightarrow$ [0,1,0].
  - **Label Encoding**: Assigns numerical values to categories based on their order. Best for ordinal data. Example: "Satisfaction" $\rightarrow$ [Poor = 0, Good = 2].

  - **Frequency Encoding**: Replaces categories with their frequency in the dataset. Example: "Product Type" → [Electronics = 50, Clothing = 30].

- **Numerical Data**
  - **Normalization (Min-Max Scaling)**: Scales data to a range $[0, 1]$. Formula:

$$X_{\text{normalized}} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

  - **Standardization (Z-score Normalization)**: Scales data to have mean 0 and std dev 1. Formula:

$$X_{\text{standardized}} = \frac{X - \mu}{\sigma}$$

  - **Binning (Discretization)**: Divides continuous data into discrete bins. Example: Age → [0-18], [19-35], [36+].

Some links for starters:

- **ML — Introduction to Data in Machine Learning** - A comprehensive overview of data types in machine learning, including structured, unstructured, and semi-structured data. *[geeksforgeeks.org]* Link
- **Data types In Machine Learning** - This resource delves into quantitative and qualitative data types, offering insights into their applications in machine learning. *[pianalytix.com]* Link
- **All you need to know about encoding techniques!** - An in-depth look at various encoding methods, such as one-hot encoding and label encoding, with practical examples. *[medium.com]* Link
- **Feature Encoding Techniques - Machine Learning** - This article discusses different encoding techniques, including label encoding and one-hot encoding, and their applications in machine learning. *[geeksforgeeks.org]* Link

5) Advanced Feature Engineering

Feature engineering is the process of selecting, transforming, and creating features from raw data to improve the performance of machine learning models. Well-engineered features can significantly enhance model accuracy and predictive power by providing meaningful representations of the data.

While some of the data processing techniques themselves fall under the purview of feature engineering, you can explore other techniques such as:

- **Polynomial Features**: Taking powers of original features to enable non-linear models.
- **Feature Combinations**: Combine two or more features in a meaningful way to get a more insightful feature. Eg. Length*Width = Area in case of housing dataset.
- Read more here.

6) Standard Machine Learning Problems and Methods

Supervised and Unsupervised Learning

There are 2 main types of Machine Learning - Supervised and Unsupervised:

- **Supervised:** Supervised machine learning involves training a model on labeled data, where each input has a corresponding output. The goal is for the model to learn the relationship between inputs and outputs so it can make accurate predictions for new, unseen data.
- **Unsupervised:** Unsupervised machine learning deals with unlabeled data, aiming to discover patterns, structures, or relationships within the data. The model isn't given explicit instructions about what to learn but instead identifies hidden patterns, such as clusters or anomalies.
- **NOTE:** In this assignment, all 8 tracks are supervised learning problems, since in all cases, one already has the target class available.

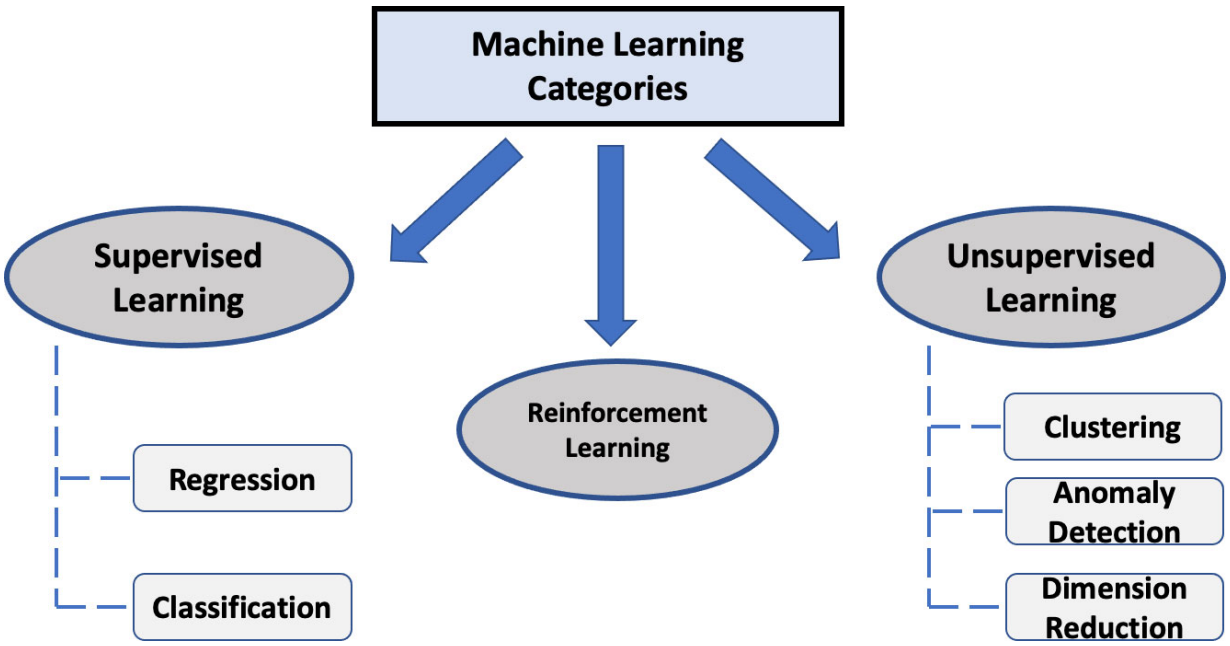

Figure 1: Diagram of Common Machine Learning Types

Some Useful Links:

- Introduction to Supervised and Unsupervised Learning
- Introduction to Classification
- Google Guide to Clustering

7) Hyperparameter Tuning

In machine learning, **model parameters** are the internal variables that a model learns from the training data. These parameters are updated during the training process to minimize the loss function, such as weights in a neural network. On the other hand, **hyperparameters** are external configurations set before the training begins, which control the learning process. Examples include learning rate, batch size, and the number of hidden layers in a neural network. For further references, read Model Parameters vs Hyperparameters.
Hyperparameter tuning is the process of finding the optimal combination of hyperparameters to improve the model's performance. This can be done using methods like grid search, random search, or more sophisticated approaches like Bayesian optimization. Proper tuning of hyperparameters can significantly enhance the accuracy and efficiency of the model. To learn more about this, you can visit here.

8) Model Selection

Model selection is the process of choosing the best model from a set of candidate models for a given dataset. This involves evaluating different models based on their performance metrics, such as accuracy, precision, recall, or mean squared error, depending on the task. The goal is to select a model that generalizes well to unseen data, balancing complexity and performance to avoid issues like overfitting or underfitting. Common techniques for model selection include cross-validation, where the dataset is split into training and validation sets to assess how well a model performs on new data. More on this could be found here.

9) Importance of Different Evaluation Metrics

While **accuracy** is a commonly used metric for evaluating models (as we will see in the assignments where we majorly use accuracy as our evaluation metric), it may not always provide a complete picture of a model's performance, especially in cases of imbalanced datasets. For instance, in scenarios where one class significantly outnumbers another, a high accuracy may simply reflect the model's ability to predict the majority class. To address such limitations, metrics like **precision**, **recall**, and the **F1-score** are often used. Precision measures the proportion of true positive predictions among all positive predictions, while recall measures the proportion of true positives among

all actual positives. The F1-score provides a balance between precision and recall, offering a more nuanced view of model performance, particularly in tasks where the cost of false positives and false negatives differs. You can learn more about these terms here.

Track 1: Bank Customer Churn Prediction

Is our customer about to leave us? Can we predict on the basis of their profile and engagement with the bank?

Dataset

The dataset can be accessed here.

Non-Competitive Part (9 Marks)

1. Exploratory Data Analysis (EDA)

1. **Understand the dataset:** Load the dataset and display its basic structures (`.head()`, `.info()`, `.describe()`).
2. **Visualise the distribution of features:** Use histograms for numerical variables and count plots for categorical features.
3. **Compare the distribution of features for each target class:** Use violin plots for numerical variables and stacked bar plots for categorical variables.

2. Data Preprocessing

1. **Remove outliers:** Use boxplots and the interquartile range (IQR) method to identify and remove outliers.
2. **One-hot encoding:** Convert categorical variables to numerical ones using one-hot encoding.
3. **Standardize numerical variables:** Apply z-score normalization using `StandardScaler` from `sklearn`.

3. Logistic Regression

1. **Data splitting:** Split the data into training and testing sets (80:20 split) without randomization.
2. **Implement Logistic Regression from scratch:** Write the algorithm from first principles and train the model.
3. **Performance evaluation:** Report accuracy, precision, recall, and F1 scores on the test set.
4. **Comparison:** Compare results with the default `sklearn` implementations of `LogisticRegression`, `SVC` and `DecisionTreeClassifier`.

Competitive Part (6 Marks)

We will be using **F1-score** as the metric to determine rankings.

1. **Feature Engineering and Selection:** Analyse correlations between features and drop highly correlated or low-variance features. Engineer new features as needed.
2. **Handle class imbalance:** Explore techniques such as oversampling (`SMOTE`), undersampling, or class-weight adjustment to address class imbalance.
3. **Experiment with advanced algorithms:** Use variations of `Decision Trees`, `SVMs`, `Bagging`, and `Boosting` to improve performance. Implement these techniques using `sklearn` and optimize hyperparameters to maximize F1-score.

NOTE: In Competitive Part, you are free to **play with the data**. The **type of models** should be restricted to as stated.

Track 2: Online Shopper Conversion Prediction

Given a person's website activity and historical analytics, can we predict whether they will make a purchase?

Dataset

The dataset can be accessed here. Refer `metadata.txt` for details about the data fields.

Non-Competitive Part (9 Marks)

2. Exploratory Data Analysis (EDA)

1. **Understand the dataset:** Load the dataset and display its basic structures (`.head()`, `.info()`, `.describe()`).
2. **Visualise the distribution of features:** Use histograms for numerical variables and count plots for categorical features.
3. **Compare the distribution of features for each target class:** Use violin plots for numerical variables and stacked bar plots for categorical variables.

2. Data Preprocessing

1. **Remove outliers:** Use boxplots and the interquartile range (IQR) method to identify and remove outliers.
2. **One-hot encoding:** Convert categorical variables to numerical ones using one-hot encoding.
3. **Standardize numerical variables:** Apply z-score normalization using `StandardScaler` from `sklearn`.

3. Logistic Regression

1. **Data splitting:** Split the data into training and testing sets (80:20 split) without randomization.
2. **Implement Logistic Regression from scratch:** Write the algorithm from first principles and train the model.
3. **Performance evaluation:** Report accuracy, precision, recall, and F1 scores on the test set.
4. **Comparison:** Compare results with the default `sklearn` implementations of `LogisticRegression`, `SVC` and `DecisionTreeClassifier`.

Competitive Part (6 Marks)

We will be using **F1-score** as the metric to determine rankings.

1. **Feature Engineering and Selection:** Analyse correlations between features and drop highly correlated or low-variance features. Engineer new features as needed.
2. **Handle class imbalance:** Explore techniques such as oversampling (`SMOTE`), undersampling, or class-weight adjustment to address class imbalance.
3. **Experiment with advanced algorithms:** Use variations of `Decision Trees`, `SVMs`, `Bagging`, and `Boosting` to improve performance. Implement these techniques using `sklearn` and optimize hyperparameters to maximize F1-score.

NOTE: In Competitive Part, you are free to **play with the data**. The **type of models** should be restricted to as stated.

Track 3: YouTube Comment Predictor

The objective of this assignment is to predict the comment count of a given YouTube video. The provided dataset (which you can download from here) includes various columns such as *channelId*, *trending_date*, *likes*, *dislikes*, etc. The target column is *comment_count_bin*, which indicates the class to which the video belongs.

Non-Competitive Part (9 Marks)

In the non-competitive part, we will focus solely on the numerical columns of the dataset, which are: *categoryId*, *view_count*, *likes*, *dislikes*, and *duration_seconds*.

1. Data Preprocessing

- Remove rows containing any NaN values in any of the columns.
- Delete the remaining columns that are not listed above.

2. Exploratory Data Analysis (EDA)

- Analyze the dataset by exploring the distribution of different features.
- Visualize the relationships between features and discuss relevant trends or patterns that could assist in predicting the comment count.
- Document your observations in the report.

3. Implement Logistic Regression

- Implement **multi-class logistic regression** from scratch using **gradient descent**, without utilizing any inbuilt machine learning libraries.
- Perform various versions of gradient descent, including stochastic gradient descent, mini-batch gradient descent, and batch gradient descent.
- Document the accuracy of each method and provide a brief explanation of your approach, detailing the optimization process and the loss function used.

4. Trying Further Models

- Train and evaluate a Decision Tree model and a Random Forest model using appropriate libraries.
- Compare the performance of these models with that of logistic regression.

Competitive Part (6 Marks)

For the competitive part, the training dataset remains unchanged, but you are free to use all columns of the dataset as desired. You may also utilize any library of your choice. **Ensure all thoughts and results are clearly documented.**

1. Enhanced Preprocessing

- Experiment with additional text preprocessing techniques, as the dataset contains text columns.
- Explore different tokenization methods.
- Parse the `trending_date` column to evaluate if it improves the model's accuracy.

2. Advanced Modelling

- Experiment with other classification techniques such as Support Vector Machines.
- Investigate any imbalances in the dataset and research methods to address them.
- Explore various NLP techniques to enhance the model's accuracy.

Track 4: Hospital Cost Prediction

In this task, we primarily deal with categorical data and aim to implement linear regression. The evaluation metric used will be Mean Squared Error (MSE), which must be minimized. The dataset is already categorized, and the target column to be predicted is *Total Costs*. The dataset can be found here.

Non-Competitive Part (9 Marks)

1. Data Preprocessing

- Remove the column *Birth Weight* due to numerous entries with a value of 0.
- Perform one-hot encoding on the categorical columns, except for *Length of Stay*, which is a numerical column.

2. Exploratory Data Analysis (EDA)

- Analyze the dataset by exploring the distribution of different features.
- Visualize the relationships between features and discuss relevant trends or patterns that could assist in predicting total costs.
- Document your observations in the report.

3. Implement Linear Regression

- Implement the closed-form solution of linear regression by solving the normal equations.
- Implement the gradient descent approach for linear regression, exploring stochastic, mini-batch, and batch gradient descent.
- Perform the above steps both before and after one-hot encoding the dataset. Document your results (MSE) appropriately.
- Compare your results with the standard scikit-learn linear regression to identify any potential errors in your implementation.

4. Lasso and Ridge Regression

- Implement Lasso and Ridge regression (built-in libraries may be used).
- Document your results, providing suitable explanations for changes in MSE values.

Competitive Part (6 Marks)

Use the same dataset as above. **Ensure all thoughts and results are clearly documented.**

- Further explore the dataset to understand which columns most significantly affect the target column.
- Experiment with different feature engineering techniques, such as adding polynomial features.
- Reintroduce the *Birth Weight* column, addressing redundancies (e.g., using the average of remaining values).
- **Important:** The primary goal is to minimize MSE without overfitting, as your code will be tested on unseen data.

Track 5 : Movie Reviews Sentiment Analysis

The goal of this assignment is to analyze movie reviews and classify them into four sentiment categories: A, B, C, and D (with 'A' being very positive sentiment and 'D' very negative sentiment). The assignment consists of both a **Non-Competitive** and a **Competitive** part.

Dataset

The dataset can be downloaded from here

Non-Competitive Part (9 marks)

1. Data Preprocessing

- Perform necessary text preprocessing steps, including:
  - Tokenization
  - Removing stop words
  - Stemming or Lemmatization
  - Any other relevant preprocessing techniques

2. Exploratory Data Analysis (EDA)

- Analyze text statistics such as:
  - Word frequency analysis
  - Sentence length distribution
  - Character count analysis, etc.
- Visualizations:
  - Word clouds
  - Histograms and bar plots to display text distributions, etc.

3. Word Embedding

- Convert text data into numerical representations using word embedding techniques.

4. Multi-Class Logistic Regression

- Implement logistic regression for multi-class classification using appropriate libraries.
- Use Stochastic Gradient Descent (SGD) and Mini-Batch Gradient Descent and compare performance.

5. Random Forest Classifier

- Train and evaluate a Random Forest model using using appropriate libraries.
- Compare its performance with logistic regression.

NOTE: Document your observations in the report for each section.

Competitive Part (6 marks)

The primary goal here is to maximize accuracy without overfitting, as your code will be tested on unseen data. Here, **instead of classes 'A', 'B', 'C' and 'D'**, you need to **classify the sentiments as classes 'AB' and 'CD'**, i.e. combine the first 2 and the last 2 classes in the classifier that you trained above.

1. Enhanced Preprocessing

- Experiment with additional text preprocessing techniques.
- Try different tokenization and vectorization methods.

2. Advanced Modeling

- While the training dataset remains unchanged, you can explore and apply more advanced machine learning models (e.g., Decision Trees, Random Forest, Gradient Boosting).
- Optimize hyperparameters and justify your choices.

NOTE: Ensure that all the enhancements and modifications attempted in this section are thoroughly documented in the report.

Track 6 : Sudoku Challenge Meter

The goal of this assignment is to predict the difficulty level of a Sudoku puzzle using machine learning techniques. You will begin with feature engineering and exploratory data analysis (EDA) before implementing regression models. The assignment consists of both a **Non-Competitive** and a **Competitive** part.

Dataset

The dataset can be downloaded from here

Non-Competitive Part (9 marks)

1. Feature Engineering

- The given dataset contains only the Sudoku puzzle itself. To build a predictive model for difficulty estimation, you need to construct relevant features.
- One example feature is **'Clues'**, which counts the number of pre-filled cells in the Sudoku grid. This is an intuitive feature because puzzles with more pre-filled numbers are generally easier to solve.
- You must create **three additional meaningful features** and provide a brief explanation of the logic behind their construction in your report.
- Redundant features (e.g., $Clues^2$, $Clues^3$, etc.) are not allowed in this part; such modifications can be explored in the **Competitive** section.

2. Exploratory Data Analysis (EDA)

- Perform an analysis on the dataset by exploring the distribution of different features.
- Visualize the relationships between features and discuss relevant trends or patterns that might help in predicting difficulty levels.
- Document your observations in the report.

3. Implementation of Linear Regression

- Implement **Linear Regression from scratch** using **Gradient Descent** (without relying on built-in machine learning libraries).
- Clearly explain your approach and optimization process in the report.

4. Lasso and Ridge Regression

- Implement **Lasso Regression** and **Ridge Regression** using appropriate libraries.
- Compare their performance with basic linear regression and mention that in the report.

Competitive Part (6 marks)

The primary goal here is to minimize MSE without overfitting, as your code will be tested on unseen data.

1. Enhanced Feature Engineering

- You can experiment with more sophisticated feature engineering techniques beyond the ones used in the **Non-Competitive** part.
- Feature transformations, additional statistical properties, and domain-specific insights are encouraged.

2. Advanced Modeling

- While the training dataset remains unchanged, you can explore and apply more advanced machine learning models (e.g., Decision Trees, Random Forest, Gradient Boosting, or Neural Networks).
- Optimize hyperparameters and justify your choices.

NOTE: Ensure that all the improvements and modifications attempted in this section are thoroughly documented in the report.

Track 7: Kingmaker

The goal of this assignment is to predict the 'Success class' (defined as 'A', 'B', 'C', or 'D', with 'A' being the most successful, and 'D' the least) of an actor, given their profile. The assignment consists of both a **Non-Competitive** and a **Competitive** part.

Dataset

The dataset for this track can be found here.

Non-Competitive Part

Note that the 'Story' column is not to be considered in this part

1. Data Pre-processing

- delete unnecessary columns (identified using EDA
- make the data more uniform (so that values for a particular feature are represented in the same way). This too requires insights from EDA.

2. Exploratory Data Analysis (EDA)

- Check the distribution of each of the classes 'A', 'B', 'C' and 'D' and identify whether there is a need to upsample/downsample.
- Check for the correlation between the target class "Success" and each of the other classes (this should be done in the form of plots). In doing so, identify which features correlate the most with the target feature.
- check the ranges of different features such as Age, Education, etc and classify which to treat as nominal data, ordinal data, discrete, or continuous data.

3. Feature Engineering

- Try various encoding methods for the categorical features
- based on the correlations, check to see if variable transformation must be applied for any of the numerical features
- remove/replace any additional words/phrases in the existing features
- find the best ways to represent the given features

4. Multi-Class Classification

- Implement the following multi-class classifiers to classify the given profile into A, B, C and D, and compare their performance:
  - KNN **(from scratch)**
  - Random forest
  - Logistic regression
- make sure to perform hyperparameter-tuning for relevant models.
- **NOTE:** Document your observations in the report for each section.

Competitive Part

- Given that the 'Story' feature is also present, generate new features (using NLP techniques) that can be constructed by extracting information from the paragraph to improve model accuracy
- In the report, detail at least two new features that are constructed, and at least provide the intuition behind each of them (through EDA or through increase in model accuracy after including the feature)
- Improve accuracy as much as possible, with freedom to use classifiers and techniques listed outside the scope of the Non-competitive part.
- compare the accuracy achieved with the NLP techniques, with that of the models employed earlier.

Ensure that all the enhancements and modifications attempted in this section are thoroughly documented in the report.

Track 8: The King, the Queen, and the all-seeing Fish

The goal of this assignment is to evaluate a chessboard given in Forsyth-Edwards Notation (FEN). One is expected to assign a numerical score to the given chessboard, such that it correlates with how useful the current state will be in winning. The assignment consists of both a **Non-Competitive** and a **Competitive** part.

Dataset

The dataset for this track can be found here.

Non-Competitive Part

1. Data Pre-processing

- create a function that re-writes the data in a simpler format.

2. Exploratory Data Analysis (EDA)

- Identify and remove outlier values for the evaluated scores.
- check the distribution of the evaluated scores
- plot graphs for different possible features of the board in comparison to the evaluated scores (for example, the number of pieces on the board vs the evaluated score, etc.)

3. Feature Engineering

- Try various encoding methods for the chess boards
- Convert the given chessboard into a format that can be best fed into a neural network
- compare the results for different formats (at least two) without the addition of more features

4. Neural Network Training and Analysis

- Train a baseline neural network with the above feature-engineered data (basically only the FEN format is encoded, no new feature columns are added per se)
- Explore different optimizers and select the one that will be the best in this case.
- make sure to perform hyperparameter-tuning.
- analyse and compare the accuracies when:
  - the number of training epochs is varied
  - the amount of training data is varied

  Using this, answer the question of whether it is better to train with limited data on multiple epochs, or with more data on fewer epochs.
- **NOTE:** Document your observations in the report for each section.

Competitive Part

- Based on EDA, find two unique new features that can be used to improve accuracy.
- Train the neural network with each of these features added, to show the improvement in accuracy
- Try using other techniques for the prediction of evaluation scores (except for pre-trained engines and models) and compare their accuracies

Ensure that all the enhancements and modifications attempted in this section are thoroughly documented in the report.

Some Relevant Links

- Guide to Stockfish Evaluator
- Forsyth-Edwards Notation

Bonus

Using the trained model, with the addition of a few basic helper functions, you can create your own chess-bot! (Guide for this will be provided once the assignment time ends, but you are more than welcome to go and explore this on your own :))

Submission Instructions

These would be released close to the assignment deadline.