Project 1 (Network Attack Prediction)

Description: Cyber adversaries are becoming more sophisticated in their efforts to avoid detection, and many modern malware tools are already incorporating new ways to bypass antivirus and other threat detection measures. Software to detect network attacks protects a computer network from unauthorized users, including perhaps insiders. In this competition, Kagglers will develop models that identify network attacks using machine learning.

Data: https://drive.google.com/drive/folders/1JvE81ivN1CsQv8rH_molTuvxAq2iiNm6

Dataset: This is a multi-class classification problem. The given dataset contains 41 feature columns and 1 label column. The task is to classify the data into 5 types of network attacks, namely:-

- ipsweep probe
- back dos
- satan probe
- portsweep probe
- normal

Consider the following details about the feature columns:-

1. duration: length (number of seconds) of the connection
2. protocol_type: type of the protocol, e.g. tcp, udp, etc.
3. service: network service on the destination, e.g., http, telnet, etc.
4. src_bytes: number of data bytes from source to destination
5. dst_bytes: number of data bytes from destination to source
6. flag: normal or error status of the connection
7. land: 1 if the connection is from/to the same host/port; 0 otherwise
8. wrong_fragment: number of ``wrong'' fragments
9. urgent: number of urgent packets
10. hot: number of ``hot'' indicators
11. num_failed_logins: number of failed login attempts
12. logged_in: 1 if successfully logged in; 0 otherwise
13. num_compromised: number of ``compromised'' conditions
14. root_shell: 1 if root shell is obtained; 0 otherwise
15. su_attempted: 1 if ``su root'' command attempted; 0 otherwise 16. num_root: number of ``root'' accesses

Evaluation: You will be evaluated based on the Mean F1 score on the test dataset. The submission format needs to be in the form specified in sample_submission.csv.

Project 2 (Used Car Price Prediction)

Description: Used Car Price Prediction Dataset is a comprehensive collection of automotive information extracted from the popular automotive marketplace website, https://www.cars.com. This dataset comprises 4,009 data points, each representing a unique vehicle listing, and includes nine distinct features providing valuable insights into the world of automobiles.

- Brand & Model: Identify the brand or company name along with the specific model of each vehicle.

- Model Year: Discover the manufacturing year of the vehicles, which is crucial for assessing depreciation and technology advancements.
- Mileage: Obtain the mileage of each vehicle, a key indicator of wear and tear and potential maintenance requirements.
- Fuel Type: Learn about the type of fuel the vehicles run on, whether it's gasoline, diesel, electric, or hybrid.
- Engine Type: Understand the engine specifications, shedding light on performance and efficiency.

- Transmission: Determine the transmission type, whether automatic, manual, or another variant.

- Exterior & Interior Colors: Explore the aesthetic aspects of the vehicles, including exterior and interior color options.
- Accident History: Discover whether a vehicle has a prior history of accidents or damage, crucial for informed decision-making.
- Clean Title: Evaluate the availability of a clean title, which can impact the vehicle's resale value and legal status.
- Price: Access the listed prices for each vehicle, aiding in price comparison and budgeting.

Data: https://drive.google.com/drive/folders/1g4NYOIE2guwxL3zfG2nym5VpJjULDeED

Used Car Prediction Data

Root Mean Squared Error of the predicted car price on the testing_data.csv file. The RMSE error will be calculated between the predicted price and the actual price of the car hidden from you. Your submission format needs to be in the form of sample_submissions.csv. Please ignore any wrong indentations, if any.

Project 3 (Flight Delay)

Description: The goal of this project is to develop a predictive model to determine whether a flight will be delayed by more than 15 minutes.

Data: https://drive.google.com/drive/folders/1MALjlqJUCL9MdvBxB6f7ab_qmc-rfg6u

Evaluation

Target Metric: ROC AUC (Receiver Operating Characteristic - Area Under Curve)

The performance of the predictive model will be evaluated using the ROC AUC metric, which measures the model's ability to distinguish between the positive class (flight delayed) and the negative class (flight not delayed).

Dataset Description

The dataset provided for this project contains both a training set and a test set, along with a sample submission file. The following CSV files are included:

- flight_delays_train.csv: This file is the training dataset used to build and train the predictive model.
- flight_delays_test.csv: This file is the test dataset used to evaluate the model's performance.

Features Included in the Dataset:

- Month: Month of the year
- DayofMonth: Day of the month
- DayOfWeek: Day of the week
- DepTime: Departure time
- UniqueCarrier: Unique carrier code of the airline
- Origin: Origin airport of the flight
- Dest: Destination airport of the flight
- Distance: Distance between the origin and destination airports
- dep_delayed_15min: Target variable indicating if a flight is delayed by more than 15 minutes (1 for delayed, 0 for not delayed)

Instructions

1. You should create a validation split and test the model’s performance and then finally do the predictions on the test set.
2. Submit Results: Prepare the final submission file in the required format and submit it for evaluation.

Project 4 (Cuisine Prediction)

Description: The objective of this assignment is to develop a machine learning model to predict the cuisine of a given recipe based on its list of ingredients. This task involves classifying the type of cuisine (e.g., Indian, Mexican, Moroccan, etc.) for each recipe in the dataset. You will be utilizing Artificial Neural Networks (ANN) to build your model, and you can choose any framework like TensorFlow or PyTorch.

Data: https://drive.google.com/drive/folders/1i0OCcSjrXXadtnd-Sro6IdNBic-b2LYU

Evaluation

For simplicity, the evaluation metric for this project is overall Accuracy:
Accuracy = (tp + tn) / (tp + tn + fp + fn)
Where:
tp: True Positives
tn: True Negatives
fp: False Positives
fn: False Negatives
Submission Format
You must submit a CSV file with exactly 10,000 entries plus a header row.
The file should have exactly 2 columns:
Id (sorted in any order)
Category (must be a string from all possible cuisines)
Example:

```
Id,Category
15717,mexican
25265,moroccan
6935,indian
46557,korean
8678,greek
```

Dataset Description
In the dataset, we include the recipe id, the type of cuisine, and the list of ingredients of each recipe (of variable length). The data is stored in JSON format.
An example of a recipe node in train.json:

```
{
"id": 24717,
"cuisine": "indian",
"ingredients": [
"turmeric",
"vegetable stock",
"tomatoes",
"garam masala",
"naan",
"red lentils",
"red chili peppers",
"onions",
"spinach",
"sweet potatoes"
]
}
```

In the test file test.json, the format of a recipe is the same as train.json, only the cuisine type is removed, as it is the target variable you are going to predict.
Instructions

1. You should create a validation split and test the model’s performance and then finally do the predictions on the test set.
2. Submit Results: Prepare the final submission file in the required format and submit it for evaluation.
dataset_link

Project 5 Robert Frost's Poem Generation

Description: Create a dataset from Robert Frost's poems and train a BiLSTM model to generate new poems using a next-word prediction strategy.

Dataset:

- Collect poems of Robert Frost; please find the dataset (https://www.gutenberg.org/cache/epub/59824/pg59824-images.html)
- Preprocess the text data by:
  - Tokenizing the words
  - Creating a vocabulary list
  - Converting text to numerical sequences
- Split the data into training and testing sets (80% for training and 20% for testing)

Model:

- Implement a BiLSTM (Bidirectional Long Short-Term Memory) model using a library like TensorFlow or PyTorch
- Configure the model with:
  - Input layer: sequence length and vocabulary size
  - Hidden layer: BiLSTM units and dropout rate
  - Output layer: softmax activation function for next-word prediction
- Train the model on the training data with:
  - Batch size and number of epochs
  - Loss function: categorical cross-entropy
  - Optimizer: Adam

Evaluation:

- Use the testing data to evaluate the model's performance
- Metrics:
  - Perplexity: measures the model's ability to predict the next word
  - Accuracy: measures the model's ability to generate coherent text
- Generate new poems using the trained model and evaluate their quality

Submission:

- Submit the trained model and generated poems
- Include a report detailing:
  - Dataset preparation and preprocessing
  - Model architecture and training parameters
  - Evaluation results and analysis
  - Generated poems and their quality assessment

*****Advanced methods (Optional): Try the above similar approach using pre-trained Transformer models (GPT and T5) and diffusion Models.**

Project 6 (Spam or Not Spam Data Generation using LSTM VAE)

Description:

Train a Long Short-Term Memory (LSTM) based Variational Autoencoder (VAE) and Diffusion model to generate synthetic spam and not spam data from the dataset available on Kaggle.

Dataset:

- Use the "Spam or Not Spam Dataset" from Kaggle (https://www.kaggle.com/datasets/ozlerhakan/spam-or-not-spam-dataset/data)
- Preprocess the text data by:
  - Tokenizing the words
  - Creating a vocabulary list
  - Converting text to numerical sequences
- Split the data into training and testing sets (80% for training and 20% for testing)

Model:

- Implement an LSTM VAE model using a library like TensorFlow or PyTorch
- Configure the model with:
  - Encoder: LSTM layers to compress input data into latent space
  - Decoder: LSTM layers to reconstruct input data from latent space
  - Latent Space: Gaussian distribution with mean and variance
- Train the model on the training data with:
  - Batch size and number of epochs
  - Loss function: reconstruction loss (mean squared error) and KL divergence
  - Optimizer: Adam or RMSprop

Evaluation:

- Use the testing data to evaluate the model's performance
- Metrics:
  - Reconstruction accuracy: measures the model's ability to reconstruct input data
  - KL divergence: measures the model's ability to learn a Gaussian distribution in latent space
- Generate synthetic spam and not spam data using the trained model and evaluate their quality

Submission:

- Submit the trained model and generated synthetic data
- Include a report detailing:
  - Dataset preparation and preprocessing
  - Model architecture and training parameters
  - Evaluation results and analysis
  - Generated synthetic data and their quality assessment

*****Advanced methods (Optional): Try the above similar approach using pre-trained Transformer models (GPT and T5) and diffusion Models.**

Project 7 (Sales Prediction)

Task description

Given a product and its corresponding details, the task is to predict the amount of sales based on the features.

Dataset: Train - 8k. The students are expected to split the dataset into train and validation as per the requirement.

Training dataset link: Click here

Columns in the dataset:

Item_Identifier: Unique identity number for a product.
Item_Weight: Indicates the weight of the product
Item_Fat_Content: Indicates the fat content- Low Fat / Regular
Item_Visibility: The percentage of total display area of a store allocated to the particular product.
Item_Type: The category to which the product belongs
Item_MRP: Maximum Retail Price (list price) of the product
Outlet_Identifier: Unique store ID
Outlet_Establishment_Year: The year in which store was established
Outlet_Size: The size of the store in terms of ground area covered
Outlet_Location_Type: The type of city in which the store is located.
Outlet_Type: Whether the outlet is just a grocery store or some sort of supermarket.
Item_Outlet_Sales: Sales of the product in the particular store. This is the outcome variable to be predicted.

Test submission format:

The test file will contain all the columns except the '**Item_Outlet_Sales**' as that is to be predicted. The students are expected to create a submission file in the following format, **[team_name]_SalesPredictionTask_submission.xlsx** (e.g. MLTitans_SalesPredictionTask_submission.xlsx) with columns names **'Item_Identifier'** and **'Item_Outlet_Sales'** containing the unique ids of the test cases and the corresponding predicted sales amount, respectively.

Evaluation

The **Root Mean Squared Error**, will be calculated between the predicted '**Item_Outlet_Sales'** and the actual value of it.

Feel free to contact us with any inquiry.

Project 8 (Disaster Classification)

Task description:

Given an image as an input, the task is to classify the image into one of the four categories: CYCLONE, EARTHQUAKE, FLOOD, and WILDFIRE.

Dataset:

	Train	**Validation**	**Test**
CYCLONE	400	100	100
EARTHQUAKE	400	100	100
FLOOD	400	100	100
WILDFIRE	400	100	100

Train and Validation data: Click here.

Note that, there is no .excel, .tsv, .csv, or .txt file containing annotated information for training and validation. Rather than the training, validation files are stored in separate sub-folders and folders with appropriate names that can be used for annotation purposes. The students are expected to curate a dataset out of the training and validation files and train a classification model.

Test submission format:

For evaluation, all the test cases will be supplied at the end. Each test case will be marked with a unique id, such as test_0, test_399, etc., under the column named 'ID'. The students are expected to create a submission file in the following format, **[team_name]_DisasterClassificationTask_submission.xlsx** (e.g., MLTitans_DisasterClassificationTask_submission.xlsx) with columns names 'ID' and 'LABEL' containing the unique ids of the test cases and the corresponding predicted labels, respectively.

For the classes, cyclone, earthquake, flood, and wildfire - use the following labels: CYCLONE, EARTHQUAKE, FLOOD, and WILDFIRE, respectively.

Evaluation

For this classification task, we will use categorical Precision, Recall, and F1 scores for each category as well as Macro Precision, Recall, and F1 scores for overall assessment.

Feel free to contact us in case of any enquiry.

Project 9 (Handwriting Recognition)

Objective: The objective of this assignment is to build and evaluate a machine learning model that can accurately recognize handwritten text using the provided dataset.

Dataset: You will use the dataset available on Kaggle using the following link: Handwriting Recognition Dataset. This dataset consists of images of handwritten characters and their corresponding labels.

The dataset contains three CSV files for test, train, and validation, and three folders containing the respective images. The CSV files have two columns, ”filename” and ”identity”. Filename is used to link an image to its identity, i.e., the text in the image. You are to use **only the train** data to train your model and the validation dataset to validate the model. Test data should not be used during training or validation.

Assignment Tasks

1. **Exploratory Data Analysis (EDA)**
   - Load the dataset and perform basic exploratory data analysis.
   - Visualize the distribution of different handwritten characters in the dataset.
   - Display a few sample images with their corresponding labels.
2. **Data Preprocessing**
   - Normalize the image data to ensure uniformity in input.
   - Split the dataset into training, validation, and testing sets.
   - Apply any necessary data augmentation techniques to improve model performance.
3. **Model Selection**
   - Choose an appropriate machine learning or deep learning model for handwriting recognition.
   - Justify your choice of model and describe the architecture (for deep learning models).
   - Implement the model using a suitable machine learning framework (e.g., TensorFlow, PyTorch, Scikit-learn).
   - Some models that you can try using are Convolutional Neural Networks in combination with Recurrent Neural Networks.
4. **Training the Model**
   - Train the model on the training data.
   - Track the training and validation accuracy/loss over epochs.
   - Implement techniques to prevent overfitting, such as dropout, early stopping, or regularization.
5. **Optimization**
   - Fine-tune the model by adjusting hyperparameters (e.g., learning rate, batch size) to improve performance.
   - Compare the results of different hyperparameter settings.
6. **Evaluation**

Evaluation will depend on the originality of your ideas, their implementation, and performance on test data. You will also be judged on how thorough you are with your data analysis and how many things you try. Keep the following points in mind:

- The model performance on the test data will be evaluated using character-wise F1 scores
- Plot the confusion matrix to analyze the model’s performance across different classes.
- Discuss any misclassifications and potential reasons for these errors.

7. **Conclusion**
   - Summarize the key findings from your experiments.
   - Discuss the effectiveness of your model and any challenges you encountered during the assignment.
   - Suggest possible improvements for future work.
8. **Submission**
   - Submit the following:
     - A Jupyter notebook (.ipynb) or Python script (.py) containing the code and explanations for each step.
     - A report summarizing your approach, findings, and conclusions (PDF format)

Project 10 (Predictive Maintenance for Industrial Equipment)

Objective: The primary objective of this project is to develop a machine learning model capable of predicting equipment failures before they occur, enabling timely maintenance and minimizing downtime. The project will focus on analyzing sensor data from industrial equipment to predict potential failures and maintenance needs.

Background: Predictive maintenance is a strategy that uses data-driven techniques to predict when equipment might fail, allowing maintenance to be performed just in time to prevent unplanned downtime. Traditional maintenance strategies, such as reactive maintenance or preventive maintenance, can be costly and inefficient. Predictive maintenance, powered by machine learning, optimizes the maintenance schedule by predicting failures based on real-time data, thereby reducing operational costs and improving equipment reliability.

Dataset: For this project, you will use the NASA CMAPSS Jet Engine Simulated Data. This dataset consists of multivariate time series data from sensors monitoring various parameters of turbofan engines, such as temperature, pressure, and vibration. The dataset is widely used for prognostics and predictive maintenance research.

Assignment Tasks

1. **Literature Review**
   - Research the current state of predictive maintenance, including traditional and machine learning approaches.
   - Explore different machine learning models commonly used in predictive maintenance, such as regression models, decision trees, random forests, and deep learning techniques like LSTM networks.
   - Identify key challenges in predictive maintenance, such as dealing with imbalanced data, feature selection, and model interpretability.
2. **Data Exploration and Preprocessing**
   - **Data Understanding:** Explore the dataset to understand its structure, including the types of sensors, the frequency of data collection, and the range of sensor values.
   - **Feature Engineering:** Extract and engineer features that could be predictive of equipment failure, such as running time, vibration trends, temperature increases, and pressure deviations.
   - **Handling Imbalanced Data:** Implement techniques to handle imbalanced datasets, such as oversampling, undersampling, or using anomaly detection methods to highlight rare failure events.
   - **Data Normalization:** Normalize or standardize the sensor data to ensure all features contribute equally to the model's predictions.
3. **Model Development**
   - **Baseline Model:** Start by developing a simple baseline model using a traditional machine learning algorithm like logistic regression or random forests to predict equipment failures.
   - **Advanced Models:** Progress to more advanced models like RNNs for sequence prediction, which can capture temporal dependencies in the data.
   - **Model Selection:** Compare the performance of different models using cross-validation and select the best-performing model based on relevant metrics.
4. **Training and Evaluation**
   - **Training:** Train the model on a labeled dataset, where the target variable indicates whether or not a failure occurred within a specific time window.
   - **Hyperparameter Tuning:** Use techniques like grid search or random search to fine-tune the hyperparameters of the chosen model.
   - **Evaluation:** Evaluate the model's performance using precision, recall, F1 score, and

ROC-AUC, with a focus on minimizing false negatives (missed failures).

5. **Conclusion**
   - Summarize the findings from the project, highlighting the most effective predictive maintenance strategies and the challenges encountered.
   - Discuss the practical implications of deploying such a system in an industrial setting, including potential cost savings and improvements in equipment reliability.
   - Suggest future research directions or improvements, such as incorporating additional data sources (e.g., environmental conditions) or exploring different machine learning algorithms.

Submission

- Submit the following:
  - Well-documented Jupyter notebook(s) (.ipynb) or Python script(s) (.py) containing the code and explanations for each step.
  - A report (PDF format) summarizing your approach, findings, and conclusions.

This project will provide practical experience in developing predictive maintenance models using machine learning, preparing you to tackle similar challenges in industrial and engineering contexts.