Classification Problem-Solving(Use all Classification algorithm and find best out of all)

Instructions:

In this assignment, you will demonstrate your understanding of all classification algorithm by solving a classification problem.

Dataset: You can use any publicly available dataset of your choice. Choose a dataset that includes categorical or numerical features and a target variable for classification.

Steps to follow:

a. Preprocess the dataset: Perform any necessary preprocessing steps such as data cleaning, managing missing values, and feature scaling.

b. Split the dataset: Split the dataset into training and testing sets. Use an appropriate ratio (e.g., 70% training, 30% testing).

c. Implement KNN algorithm: Implement the KNN algorithm using a Python. You can use libraries such as scikit-learn to facilitate the implementation.

d. Train the model: Train all classifiers model on the training data.

e. Predict and evaluate: Use the trained model to make predictions on the testing data. Evaluate the model's performance by calculating relevant classification metrics such as accuracy, precision, recall, and F1 score.

f. Experiment and optimize: Experiment with different values of K and evaluate the impact on the model's performance. Optimize the model by selecting the optimal value of K based on the evaluation results.

g. Document your findings: Provide a detailed explanation of the steps taken, the observations made, and the results. Include visualizations, if applicable, to support your analysis.

Submission Guidelines:

a. Submit your assignment as a pdf, and video file that includes the following sections:

Introduction: Briefly explain the purpose and objective of the assignment.

Dataset: Provide details about the dataset used, including its source and a description of the features and target variable.

Methodology: Describe the preprocessing steps, data splitting, implementation of each algorithm, and model evaluation techniques used.

Results: Present the results of your analysis, including the performance metrics obtained and any visualizations created.

Discussion: Discuss your observations, insights, and any challenges encountered during the assignment.

b. Include the source code of your implementation as an appendix or provide a link to a code repository if applicable.

Object Detection with Detectron or YOLO

Instructions:

In this assignment, you will demonstrate your understanding of object detection using either the Detectron or YOLO (You Only Look Once) framework.

Dataset: Select a dataset suitable for object detection tasks. You can use publicly available datasets such as COCO (Common Objects in Context) or any other dataset that includes annotated images for object detection.

Steps to follow:

a. Install and set up the chosen framework: Install and set up either Detectron or YOLO framework based on your preference. Follow the official documentation or tutorials to set up the framework on your local machine or preferred development environment.

b. Preprocess the dataset: Preprocess the dataset to ensure it is compatible with the chosen framework. This may involve converting the annotations to the required format and organizing the images and annotations accordingly.

c. Train the model: Use the chosen framework to train an object detection model on the dataset. Experiment with different architectures, hyperparameters, and training strategies to optimize the model's performance.

d. Evaluate the model: Once the model is trained, evaluate its performance on a separate validation or test set. Calculate metrics such as mean Average Precision (mAP) or Intersection over Union (IoU) to assess the accuracy and quality of the object detection results.

e. Fine-tune the model: Experiment with techniques such as transfer learning or fine-tuning to further improve the model's performance. Fine-tune the model using additional annotated data or by adjusting hyperparameters.

f. Visualize the results: Use the trained model to detect objects in sample images from the dataset. Visualize the detection results by overlaying bounding boxes and class labels on the images.

g. Document your findings: Provide a detailed explanation of the steps taken, the observations made, and the results. Include visualizations, if applicable, to support your analysis.

Introduction: Submit your assignment as a pdf, and video file that includes the following sections

Dataset: Describe the dataset used, including its source, size, and annotation format.

Methodology: Explain the steps followed, such as data preprocessing, model training, evaluation, and any fine-tuning techniques applied.

Results: Present the evaluation results, including metrics and visualizations of object detection outputs.

Discussion: Analyse and discuss the performance of the model, limitations, and potential areas for improvement.

Conclusion: Summarize the key findings and lessons learned from the assignment.

h. Include any relevant code snippets, configuration files, and visualization outputs as appendices or provide links to a code repository if applicable.

Sentiment Analysis using NLP

Instructions:

In this assignment, you will demonstrate your understanding of sentiment analysis using Natural Language Processing (NLP) techniques.

Dataset: Select a dataset suitable for sentiment analysis tasks. You can use publicly available datasets such as IMDb movie reviews, Twitter sentiment analysis, or any other dataset that includes text data labelled with sentiment (positive, negative, or neutral).

Steps to follow:

a. Dataset preprocessing: Preprocess the dataset by cleaning the text, removing any noise or irrelevant information, and performing any necessary text normalization techniques such as tokenization, stemming, or lemmatization.

b. Split the dataset: Split the dataset into training and testing sets. Use an appropriate ratio (e.g., 70% training, 30% testing).

c. Feature extraction: Convert the textual data into numerical features that can be used by machine learning models. Use techniques such as bag-of-words, TF-IDF (Term Frequency-Inverse Document Frequency), or word embeddings (e.g., Word2Vec, GloVe).

d. Model selection and training: Choose a suitable machine learning or deep learning model for sentiment analysis, such as Naive Bayes, Support Vector Machines (SVM), Recurrent Neural Networks (RNN), or Transformers (e.g., BERT). Train the selected model on the training dataset.

e. Model evaluation: Evaluate the trained model on the testing dataset by calculating relevant evaluation metrics such as accuracy, precision, recall, and F1 score. Assess the model's performance in predicting sentiment labels.

f. Fine-tuning and optimization: Experiment with hyperparameter tuning and model optimization techniques to improve the performance of the sentiment analysis model. Explore techniques such as grid search, cross-validation, or ensemble methods to enhance the model's effectiveness.

g. Prediction and analysis: Use the trained model to predict the sentiment labels of new, unseen text samples. Analyse the model's predictions and assess its strengths and limitations in handling different sentiment expressions and contexts.

g. Document your findings: Provide a detailed explanation of the steps taken, the observations made, and the results. Include visualizations, if applicable, to support your analysis.

Introduction: Submit your assignment as a pdf, and video file that includes the following sections

Dataset: Describe the dataset used, including its source, size, and sentiment label distribution.

Methodology: Describe the preprocessing techniques, feature extraction methods, model selection, and training strategies employed.

Results: Present the evaluation metrics, including accuracy and other relevant performance indicators. Discuss the model's strengths and weaknesses.

Discussion: Analyse the results, highlight any challenges faced during the assignment, and propose improvements or future directions.

Conclusion: Summarize the key findings and lessons learned from the assignment.

i. Include any relevant code snippets, configuration files, and visualizations as appendices or provide links to a code repository if applicable.

Time Series Analysis and Forecasting

Instructions:

In this assignment, you will demonstrate your understanding of time series analysis and forecasting techniques.

Dataset: Select a time series dataset suitable for analysis and forecasting. You can use publicly available datasets such as stock prices, weather data, or any other dataset that exhibits a time-dependent pattern.

Steps to follow:

a. Dataset exploration: Explore the time series dataset by analysing its properties, such as trend, seasonality, and noise. Visualize the data using line plots, histograms, or other appropriate techniques.

b. Preprocessing: Preprocess the time series data by managing missing values, smoothing noisy data, and addressing any anomalies or outliers. Apply techniques such as interpolation, filtering, or data transformation, as necessary.

c. Time series decomposition: Decompose the time series into its trend, seasonality, and residual components. Use decomposition techniques such as moving averages, seasonal decomposition of time series (STL), or other appropriate methods.

d. Model selection: Select an appropriate time series forecasting model based on the characteristics of the data. Consider models such as ARIMA (Autoregressive Integrated Moving Average), SARIMA (Seasonal ARIMA), Exponential Smoothing, or Prophet.

e. Model training and validation: Split the time series dataset into training and testing sets. Train the selected model on the training data and evaluate its performance using appropriate metrics such as mean absolute error (MAE) or root mean squared error (RMSE).

f. Forecasting: Use the trained model to make future predictions and generate forecasts for the time series. Visualize the forecasts along with the actual data to assess the model's accuracy and reliability.

g. Parameter tuning and optimization: Experiment with different model configurations and hyperparameter settings to optimize the forecasting performance. Explore techniques such as grid search, cross-validation, or model ensembles to enhance the accuracy of the forecasts.

h. Documentation: Provide a comprehensive report or document that includes the following sections:

Introduction: Submit your assignment as a pdf, and video file that includes the following sections

Dataset: Describe the time series dataset used, including its source, length, and any key features.

Methodology: Describe the steps followed, such as data preprocessing, time series decomposition, model selection, training, and validation.

Results: Present the evaluation metrics, including forecast accuracy measures and visualizations of the predictions. Discuss the strengths and limitations of the forecasting model.

Discussion: Analyse the results, highlight any challenges encountered during the assignment, and propose improvements or future directions.

i. Include any relevant code snippets, configuration files, and visualizations as appendices or provide links to a code repository if applicable.

Extra Additional Question : Solve M5 forecasting use case for demand forecasting
[M5 Forecasting - Accuracy | Kaggle]

Extra Additional Question : Solve Customer Churn Analysis use case.
link : https://www.kaggle.com/datasets/ankitverma2010/ecommerce-customer-churn-analysis-and-prediction

Note: Students can attempt one or two additional questions after solving the above four problems.
Note: Separate video presentations and PDFs are required for each use case.

File Format : -

```
Name_StudentID /
 - Assignment1/
   - _Assignment1.pdf (inside the PDF file, include a link to the video file)
   - Assignment1.mp4
 - Assignment2/
   - Assignment2.pdf  (inside the PDF file, include a link to the video file)
   - Assignment2.mp4
```