UNIVERSIDADE da MADEIRA

Data Science (Ciência de Dados)

Project

Part 1 (first three phases of the data analytics lifecycle) to be programmed in Python and R:

1. **Problem Formulation (phase 1):**
   - Clearly define the problem that the dataset addresses.
   - Specify the goals and objectives of the data analysis.
2. **Data Analysis and Clensing (phase 2):**
   - **Pre-processing**
     1. Describe the dataset, its source, and any preprocessing steps taken.
     2. Include data cleansing and normalization/standardization.
   - **Exploratory Data Analysis (EDA):**
     1. Conduct descriptive statistics and visualizations to understand the data.
     2. Use standard statistical methods (such as histograms) to identify patterns, outliers, and correlations.
     3. Use dimension reduction methods to identify patterns in the data (use at least one linear and one non-linear method, for example, PCA and UMAP).
     4. Discuss any initial insights gained from EDA.
   - **Hypothesis Testing:**
     1. Formulate null and alternative hypotheses.
     2. Choose appropriate statistical tests.
     3. Perform hypothesis testing and interpret the results.
3. **Model Selection (phase 3):**
   - Conduct feature engineering (produce at least 10 new features).
   - Initiate model selection, examining the set of models that would be suitable for the problem at hand.
   - Asses the most suitable model validation method.
   - Justify all the analysis.

Part 2 – Implementing Machine Learning Models (the last three phases of the data analytics lifecycle) to be programmed in Python.

1. **Model building (phase 4):**
    - **kNN algorithm:**
        1. Implement the kNN algorithm from scratch using only numpy arrays with a suitable data structure.
        2. Provide a detailed explanation and documentation of the produced kNN algorithm.
        3. Apply the implemented algorithm to the dataset and evaluate its performance.
        4. Discuss the results.
    - **Supervised Learning:**
        1. Test at least two supervised learning models (other than kNN) using the sklearn library.
        2. Discuss the results.
    - **Ensemble model:**
        1. Choose two suitable ensemble models, one with bagging and one with boosting, and apply them to the problem.
        2. Discuss the results.
    - **Deep Learning Model:**
        1. Choose a deep learning architecture (either implement layer-by-layer or employ transfer learning) and implement the deep learning model using TensorFlow or PyTorch.
        2. Discuss the results.
    - **Clustering:**
        1. Apply at least two clustering algorithms and vary the number of clusters to assess the presence of patterns in the data.
        2. Discuss the results.
2. **Model Comparison and Evaluation (phase 5):**
    - Compare the performance of the models in a table with suitable representation.
    - Use appropriate evaluation metrics.
    - Discuss the strengths and weaknesses of each approach.
    - Summarize the findings and insights from the entire project.
    - Provide recommendations for further improvements or actions.
3. **Operationalize (phase 6):**
    - Produce the final report and technical document describing the code.
    - Prepare the presentation.
    - Discuss how to implement the models in a production environment (deployment plan).

Deliverables:

1. **Mid-journey Report (delivered in Parts 1):**
    - Detailed report with a justification for all actions and discussion of the insights provided by data.
2. **Final Report (delivered in Parts 2):**
    - A comprehensive report documenting each phase, Including code snippets, visualizations, and interpretations (the document should also summarize lessons learned during the project, including challenges faced, unexpected findings, and insights gained).
3. **Executive Summary (delivered in Parts 2):**
    - A condensed version of the final report (two pages), suitable for non-technical stakeholders, highlighting the key findings, insights, and recommendations concisely.
4. **Code:**
    - Submit R and Python code, clearly commented for understanding with proper documentation. For Python, use procedural style (using functions for most calculations) or object-oriented programming.
        1. **Data Exploration and Preprocessing Notebook (delivered in Part 1):** A Jupyter notebook or equivalent detailing the exploration and preprocessing steps applied to the raw data; this should include data cleasing, feature engineering, and any transformations performed on the dataset.
        2. **Modeling Notebook (delivered in Part 2):** A notebook that outlines the modeling process, including the choice of algorithms, hyperparameter tuning, cross-validation results, and the final model's performance metrics; should be accompanied by visualizations or plots illustrating the model's behavior.
        3. **Codebase (delivered in Part 2):** The codebase containing scripts or notebooks used throughout the project, to ensure that the analysis and modeling processes are reproducible and can be easily shared or transferred to other team members.
        4. **Trained Models (delivered in Part 2):** The final trained models should be saved and documented to allow possible future deployment in a production environment or for further analysis (provide the models in a zip folder).
        5. **Visualization Dashboard (optional, delivered in Part 2):** An interactive dashboard or visualization tool allowing stakeholders to visually explore the results.
        6. **Documentation (delivered in Parts 1 and 2):** Comprehensive documentation for the code, models, and any other relevant components to facilitate future maintenance, collaboration, and knowledge transfer.
5. **Final Presentation and Defense:**
    - Each group should prepare a concise 15-minute presentation summarizing key findings and methodologies. Followed by a 15-minute defence.

Note 1: All reports must be done using a LaTeX editor (Overleaf is recommended) using either IEEE double-column style or IEEE one-column style.
Note 2: All code should be maintained in a private GitHub repository for version control.
Note 3: Each group comprises two students (can also be just one), and each group member should know the implementation and the developed code in detail for the defence.

Specification:

- For this project, use the Flight Delay and Cancellation Dataset (2019–2023), available on Kaggle (https://www.kaggle.com/datasets/patrickzel/flight-delay-and-cancellation-dataset-2019-2023/data). It is a dataset derived from the U.S. Department of Transportation's On-Time Performance Reporting System. It contains information about commercial domestic flights operated by U.S. airlines, covering millions of flight records per year.
    - Dataset description:
        - Each row in the dataset corresponds to a single scheduled commercial flight. The columns are:
            - Flight-related:
                - FL_DATE: Date of the flight (YYYY-MM-DD).
                - AIRLINE: Airline name.
                - AIRLINE_DOT: Airline name standardized by the U.S. Department of Transportation.
                - AIRLINE_CODE: Two-letter IATA airline code.
                - DOT_CODE: Unique DOT identifier for the airline.
                - FL_NUMBER: Flight number assigned by the airline.
                - ORIGIN: IATA airport code of the departure airport.
                - ORIGIN_CITY: City and state of the departure airport.
                - DEST: IATA airport code of the destination airport.
                - DEST_CITY: City and state of the destination airport.
            - Scheduled and actual times:
                - CRS_DEP_TIME: Scheduled departure time.
                - DEP_TIME: Actual departure time.
                - CRS_ARR_TIME: Scheduled arrival time.
                - ARR_TIME: Actual arrival time.
            - Delay metrics:
                - DEP_DELAY: Departure delay in minutes (actual departure minus scheduled departure). Negative values indicate early departure.
                - ARR_DELAY: Arrival delay in minutes (actual arrival minus scheduled arrival). This variable is commonly used to define whether a flight is delayed.
                - TAXI_OUT: Time (in minutes) from gate departure to wheels-off.
                - TAXI_IN: Time (in minutes) from wheels-on to arrival at the gate.
                - WHEELS_OFF: Time when the aircraft leaves the ground.
                - WHEELS_ON: Time when the aircraft touches down.
            - Flight duration and distance:
                - CRS_ELAPSED_TIME: Scheduled total flight time (minutes).
                - ELAPSED_TIME: Actual total flight time (minutes).
                - AIR_TIME: Time spent in the air (minutes).
                - DISTANCE: Distance between origin and destination airports (miles).
            - Flight Status Indicators:

        - CANCELLED: Binary indicator (1 if the flight was cancelled, 0 otherwise).
        - CANCELLATION_CODE: Reason for cancellation (e.g., weather, carrier, airspace).
        - DIVERTED: Binary indicator (1 if the flight was diverted to another airport).
    - Causes of delay (in minutes), these fields are typically populated when the arrival delay is 15 minutes or more:
        - DELAY_DUE_CARRIER: Delay caused by airline-related issues.
        - DELAY_DUE_WEATHER: Delay caused by weather conditions.
        - DELAY_DUE_NAS: Delay due to the National Airspace System (e.g., congestion).
        - DELAY_DUE_SECURITY: Delay caused by security issues.
        - DELAY_DUE_LATE_AIRCRAFT: Delay due to the late arrival of the aircraft from a previous flight.
- Project Objectives
    - Regression:
        - Predict arrival delay duration (in minutes) for a given flight based on scheduled times, airline, route, distance, and operational factors.
    - Classification:
        - Reformulate the delay prediction task as a classification problem with 3 classes:
            - On-time: Arrival delay < 15 minutes
            - Short delay: 15–30 minutes
            - Long delay: > 30 minutes
    - Clustering:
        - Identify patterns in operational performance by clustering (optional to do both, can do just one):
            - Airports based on delay behavior and traffic characteristics.
            - Airlines based on punctuality, delay causes, and route profiles.
    - Hypothesis testing:
        - Test statistically hypotheses, such as (these are just examples that can be used):
            - “Certain airlines are systematically delayed compared to others.”
            - “Weather-related factors have a statistically significant impact on flight delays.”
- **Note on data leakage:**
    - To ensure the validity and fairness of the predictive models developed in this project, pay special attention to sources of data leakage.
    - Columns that must not be used as predictors (because they either directly encode the target variable, are derived from it, or contain post-event information:
        - Delay Target and Derived Variables
        - ARR_DELAY (target variable for regression and classification)
        - DEP_DELAY (directly related to ARR_DELAY and leaks delay information)
        - DELAY_DUE_CARRIER

        - DELAY_DUE_WEATHER
        - DELAY_DUE_NAS
        - DELAY_DUE_SECURITY
        - DELAY_DUE_LATE_AIRCRAFT
        - ARR_TIME (actual arrival time)
        - DEP_TIME (actual departure time)
        - WHEELS_OFF
        - WHEELS_ON
        - TAXI_OUT
        - TAXI_IN
        - ELAPSED_TIME
        - AIR_TIME
    - Cancelled or diverted flights do not have meaningful arrival delay values and should be either excluded from delay prediction tasks or treated as separate classification problems (optional for this project)
    - The following types of variables are generally acceptable, as they are known before departure:
        - Temporal information:
            - FL_DATE
            - CRS_DEP_TIME
            - CRS_ARR_TIME
            - Derived features (hour of day, day of week, month, season)
        - Airline and route information:
            - AIRLINE, AIRLINE_CODE, AIRLINE_DOT
            - ORIGIN, DEST
            - ORIGIN_CITY, DEST_CITY
            - FL_NUMBER (likely not relevant)
        - Flight characteristics:
            - DISTANCE
            - CRS_ELAPSED_TIME