Task 1  
  
Requirements

- Encode data

- Handle missing values if any

- Correct errors, inconsistencies, remove duplicates if any

- Outlier detection and treatment if any

- Normalization / Standardization if necesarry

- Feature engineering

- Train test split, save it.

- Others?

Deliverable:

- Notebook code with no errors.

- Preprocessed data as csv.

Task 2

Requirements

- Analyze hours-per-week Target Distribution

    - Provide visualizations like histograms or density plots for hours-per-week to assess its distribution, outliers, and skewness.

- Provide a summary of the dataset using descriptive statistics, such as mean, median, and standard deviation.

- Check for missing values and report the number and percentage of missing data for each column.

- Plot histograms for at least three numerical columns and discuss their distributions, such as skewness and modality.

- Generate bar plots for at least three categorical columns and discuss the frequency distributions.

- Identify outliers in at least one numerical column using boxplots.

- Create a correlation matrix for target hours-per-week and the rest of numerical variables and discuss the strongest and weakest correlations with the target.

- Explore relationships between at least one numerical variable and hours-per-week target using scatter plots and identify any visible trends. Choose the variable that is/are most correlated with hours-per-week.

- Analyze the relationship between hours-per-week and all categorical variable using boxplots or violin plots.

- Write a summary report of your findings from the EDA, highlighting key patterns or trends, unexpected insights or anomalies, and areas requiring further investigation.

- At the end, do the same analysis for the preprocessed dataset (just change the input data in the EDA), and write a report. - Optional

- Others? - Optional

Deliverable:

- Notebook code with no errors.

- Include all visuals from the requirements in the notebook: graphs, plots, histograms, heatmap etc

- Be sure to discuss the findings and add a summary report

Task 3

Requirements

- Create a regression model on the Census dataset, with 'hours-per-week' target

- You can use models (estmators) from sklearn, but feel free to use any library for traditional ML.

    - Note: in sklearn, the LinearRegression estimator is based on OLS, a statistical method. Please use the SGDRegressor estimator, since this is based on gradient descent.

    - You can use LinearRegression estimator, but only as comparison with the SGDRegressor - Optional.

- Model Selection and Setup:

    - Implement multiple models, to solve a regression problem using traditional ML:

        - Linear Regression

        - Decision Tree Regression

        - Random Forest Regression - Optional

        - Ridge Regression - Optional

        - Lasso Regression - Optional

    - Choose a loss (or experiment with different losses) for the model and justify the choice.

        - MSE, MAE, RMSE, Huber Loss or others

    - Justify model choices based on dataset characteristics and task requirements; specify model pros and cons.

- Data Preparation

    - Use the preprocessed datasets from Task 1.

    - From the train set, create an extra validation set, if necesarry. So in total there will be: train, validation and test datasets.

    - Be sure all models have their data preprocessed as needed. Some models require different, or no encoding for some features.

- Model Training and Experimentation

    - Establish a Baseline Model:

        - For each model type, train a simple model with default settings as a baseline.

        - Evaluate its performance to establish a benchmark for comparison.

    - Make plots with train, validation loss and metric on epochs (or on steps), if applicable. - Optional

    - Feature Selection:

        - Use insights from EDA in Task 2 to identify candidate features by analyzing patterns, relationships, and distributions.

    - Experimentation:

        - For each baseline model type, iteratively experiment with different combinations of features and transformations.

        - Experiment with feature engineering techniques such as interaction terms, polynomial features, or scaling transformations.

        - Identify the best model which have the best performance metrics on test set.

    - Hyperparameter Tuning - Optional:

        - Perform hyperparameter tuning only on the best-performing model after evaluating all model types and experiments.

        - Consider using techniques like Grid Search for exhaustive tuning, Random Search for quicker exploration, or Bayesian Optimization for an intelligent, efficient search of hyperparameters.

        - Avoid tuning models that do not show strong baseline performance or are unlikely to outperform others based on experimentation.

        - Ensure that hyperparameter tuning is done after completing feature selection, baseline modeling, and experimentation, ensuring that the model is stable and representative of the dataset.

- Model Evaluation

    - Evaluate models on the test dataset using regression metrics:

        - Mean Absolute Error (MAE)

        - Mean Squared Error (MSE)

        - Root Mean Squared Error (RMSE)

        - R² Score

    - Choose one metric for model comparison and explain your choice

    - Compare the results across different models. Save all experiment results into a table.

Feature Importance - Optional

- For applicable models (e.g., Decision Tree Regression), analyze feature importance and discuss its relevance to the problem.

Deliverables

- Notebook code with no errors.

- Code and results from experiments. Create a table with all experiments results, include experiment name, metrics results.

- Explain findings, choices, results.

- Potential areas for improvement or further exploration.