Deliverable 2: Data Preparation, Modeling, Evaluation

Students Health🩺 & Academic📖 Performance🎗️

Group 14

Sowmya Kanjula
Shubh Almal
Venkata Mahalakshmi Vishnumolakala
Mounika Muddy Paka

Data Preparation

Data Cleaning:

1. **Handling Missing Data:**
    - Identified missing values and decided on an appropriate strategy for each instance. This included:
        - **Imputation:** Filling in missing values with statistical measures (mean, median, mode) or using more sophisticated methods like K-Nearest Neighbors (KNN) imputation.
        - **Deletion:** Removing records or features with a high proportion of missing values if they cannot be reliably imputed.
2. **Removing Duplicates:**
    - Scanned the dataset for duplicate records and removed them to ensure that each data entry was unique, thereby preventing redundancy and potential biases in the analysis.
3. **Modifying Data Types:**
    - Ensured consistency in data types across the dataset. This involved:
        - Converting numerical data stored as strings to appropriate numerical types (integers or floats).
        - Standardizing date formats.
        - Ensuring categorical variables were correctly identified and encoded.

Data Transformation:

1. **Scaling and Normalization:**
    - Adjusted the scale of the numerical features to ensure uniformity. This was done using methods such as:
        - **Min-Max Scaling:** Transforming the features to a fixed range, usually [0, 1].
        - **Standardization:** Centering the features by subtracting the mean and scaling to unit variance.
2. **Encoding Categorical Variables:**
    - Converted categorical variables into numerical values using techniques like:
        - **One-Hot Encoding:** Creating binary columns for each category.
        - **Label Encoding:** Assigning each category a unique integer value.
3. **Feature Engineering:**
    - Created new features from existing ones to capture more information. This included:
        - Generating interaction terms (multiplying or adding features).
        - Creating polynomial features.
        - Extracting date-time features (e.g., day, month, year, weekday).

Data Integration:

- **Merging Data:**
    - Integrated data from various sources ensuring consistent identifiers (e.g., merging datasets on a common ID).
    - Resolved conflicts and discrepancies between datasets, maintaining data integrity and coherence.

Data Splitting:

- **Training and Testing Sets:**
    - Split the dataset into training and testing sets to evaluate the model's performance. Common splits include:
        - 80% training and 20% testing.
        - 70% training and 30% testing.

Feature Selection:

- **Attribute Selection:**
    - Chose the most relevant features for model building through:
        - **Correlation Analysis:** Examined the correlation between features and the target variable to identify significant predictors.

- **Domain Knowledge:** Leveraged expertise in the subject area to select meaningful features.
- **Feature Importance Measures:** Used techniques like Random Forest Importance, LASSO, or Recursive Feature Elimination (RFE) to rank and select key features.

Modeling: two or more methods (be sure to explore the use of Pycaret)

1. Data Preprocessing

Explanation: Preprocessing Data Before developing models, it's important to clean and prepare the data for analysis. This includes:

- **Handling missing values:** Choose whether to fill in missing values or delete rows/columns with missing data.
- **Encoding categorical variables:** Encoding categorical variables One-hot encoding is a technique for converting category data into numerical format.
- **Normalizing or scaling numerical features:** To normalize or scale numerical features: Adjust the numerical data range to ensure that all features provide an equal contribution to the model.
- **Splitting the data:** Splitting the data divides the dataset into training and testing sets to assess the model's performance on new data.

2. Exploratory data analysis (EDA)

Explanation: EDA improves data interpretation by discovering patterns, anomalies, and correlations. This includes:

- Histograms, box plots, and other visual tools are used to visually represent data distribution.
- Use heatmaps or correlation matrices to identify correlations between features.
- Key statistics are summarized to get insight into the central tendency and variability of the data.

3. Modeling using traditional machine learning methods

Start with two classic machine learning models, such as Decision Trees and Random Forests, to see how they perform on your dataset.

Decision Trees:

- A decision tree divides the data into branches and makes decisions based on feature values.
- Train the model with your training data.
- Analyze the model's performance using metrics such as accuracy, precision, recall, and F1-score.

Random Forest:

- A random forest is an ensemble method that improves performance by combining several decision trees.
- Using the data you've collected, train the model.
- Assess the model's performance using the same metrics as the decision tree.

4. Modeling with PyCaret:

Explanation: PyCaret, a low-code machine learning toolkit, automates the modeling process, making it faster to develop and compare models.

Setup: Import your dataset into PyCaret and define the target variable.

Compare Models: PyCaret will train and evaluate numerous models for you to compare their performance.

Model Creation and Tuning: You can increase performance by creating particular models and fine-tuning their hyperparameters.

Model Evaluation: Use PyCaret's visualization tools to assess the performance of the models.

5. Comparison of Models

Explanation: Compare the performance of standard and PyCaret-trained models to get the best fit for your data.

Decision Tree and Random Forest: To understand the strengths and shortcomings of decision trees and random forests, evaluate them with classification metrics and confusion matrices.

PyCaret Models: Models for PyCaret Compare the different models trained by PyCaret and utilize visual tools to evaluate their performance.

Summary: By experimenting with various modeling strategies, including traditional machine learning methods and automated tools like PyCaret, you may find the best

successful model for your dataset. This complete strategy guarantees that you evaluate all views and make informed decisions based on model performance.

Evaluation: (which method provided the most accuracy/best results)

Evaluation of Models

When assessing machine learning models, a few measurements and strategies can be utilized to decide their execution and reasonableness for the errand. Here, we'll talk about the assessment handle, centering on accuracy, precision, recall, F1-score, and AUC-ROC.

1. Accuracy

- **Definition**: Accuracy is the number of correct guesses compared to the total number of guesses. It checks how accurately the model works.
- **Importance**: Helpful when the classes are equal, providing an easy way to see how well the model works..
- **Example**: If a model gets 90 answers right out of 100, it is 90% accurate.

2. Precision

- **Definition**: Precision is how many times the model correctly predicts positive outcomes compared to all the positive predictions it made.
- **Importance**: Shows how many of the predicted positive cases are really positive, which helps to lower the number of false positives.
- **Example**: if a model finds 10 true positives and 8 of them are right, the precision is 80%.

3. Recall (Sensitivity)

- **Definition**: Recall is the number of correct positive predictions compared to all the actual positives.
- **Importance**: It shows how good the model is at identifying all the positive cases and decreasing missed positives.
- **Example**: if there are 20 real positive cases and the model correctly finds 15 of them, the recall is 75%.

4. F1-Score

- **Definition**: The F1-score is a way to combine precision and recall into one number that shows how well a model is doing. It balances both aspects equally.

- **Importance**: Helpful when the data has uneven groups, giving a better idea of performance than just accuracy.
- **Example**: if accuracy is 80% and completeness is 70%, the F1-score is about 74%.

5. AUC-ROC (Area Under the Receiver Operating Characteristic Curve)

- **Definition**: AUC-ROC shows how well a model can tell the difference between different groups. It does this by comparing the rate of true positives to the rate of false positives on a graph.
- **Importance**: Shows an overall measure of how well the model performs at different decision points, indicating how well it can tell apart different classes.
- **Example**: An AUC-ROC score of 0. 9 means the model is 90% likely to correctly tell the difference between positive and negative groups.

Choosing the Best Model

- **Pycaret**: Pycaret has tools that make it easy to set up and compare different models, helping you quickly find the best ones based on different measures.
- **Manual Models**: By working with models like Logistic Regression and Random Forest by hand, you can adjust settings and understand how the models work better..

Conclusion/Results: (what did you learn)

In this project, we undertook a comprehensive approach to data preparation and modeling, starting with meticulous data cleaning, including handling missing values through imputation and deletion, removing duplicates, and ensuring consistent data types. We transformed the data by scaling numerical features and encoding categorical variables, and performed feature engineering to enhance the dataset's predictive power. Integration of various data sources and thoughtful data splitting ensured robust model evaluation. We employed traditional machine learning models, such as Decision Trees and Random Forests, and leveraged PyCaret for automated model building and comparison. Evaluation metrics, including accuracy, precision, recall, F1-score, and AUC-ROC, provided a detailed assessment of model performance. Overall, the project highlighted the importance of thorough data preparation, diverse modeling techniques, and comprehensive evaluation, while also identifying potential issues with imputation, feature engineering, and model interpretability.

Known Issues (problems with predictors, reporting, bias, etc.)

1. **Handling Missing Data -** Imputation, while effective, can introduce bias if the missing data is not random. This can affect model performance and generalizability. And deleting records or features with a high proportion of missing values can lead to loss of valuable information, potentially impacting the model's predictive ability.
2. **Feature Engineering and Selection -** Generating interaction terms and polynomial features can lead to multicollinearity, where features become highly correlated, affecting model stability and interpretability. Relying heavily on feature importance measures without sufficient domain knowledge can result in overlooking important predictors or including irrelevant ones.
3. **Modeling Techniques -** Traditional models like Decision Trees and Random Forests can be prone to overfitting, especially with high-dimensional data or complex feature interactions.
4. **Evaluation Metrics -** Accuracy alone is not sufficient for evaluating models, especially in cases of imbalanced classes. Precision, recall, F1-score, and AUC-ROC provide a more nuanced view of model performance. There is a risk of over-reliance on automated tools for model evaluation, potentially missing out on domain-specific insights and nuances that manual tuning and evaluation might reveal.