EDA and Machine Learning Notes

1. Understanding the Dataset

- Before doing any processing, it’s essential to understand the dataset’s structure, content, and potential problems.

- Loading the Data: Use pandas.read_csv() for CSV files. For Excel, use read_excel(), and for JSON use read_json(). Loading the dataset correctly ensures we start with the right format.

- Previewing Data: Use .head() to check the first few rows for column names and sample values. Use .tail() to check the last few rows for potential issues like missing values at the end.

- Shape and Structure: .shape returns (rows, columns) so you know the dataset’s size. .info() shows column names, data types, and the number of non-null entries, which helps in spotting missing values.

- Summary Statistics: .describe() gives statistical information like mean, standard deviation, and quartiles, useful for detecting unusual values or scaling needs.

2. Handling Missing Values

- Missing data can cause bias in your analysis and errors in model training.

- Identify Gaps: df.isnull().sum() counts missing values in each column so you can target only affected areas.

- Removing Missing Data: Use dropna() when the missing values are too many to fill meaningfully, or when they occur in rows/columns that aren’t critical.

- Filling Missing Values: Use fillna() when data is important but incomplete. Mean and median are common for numeric data; mode or a fixed value is better for categorical data.

3. Data Type Conversion

- Correct data types ensure accurate analysis and prevent errors in calculations or transformations.

- Converting Columns: .astype() is used to change a column to integer, float, or string types as required by the model or analysis.

- Handling Dates: Dates stored as strings limit analysis. Convert them to datetime using pd.to_datetime() to enable time-based filtering, grouping, and plotting.

4. Detecting and Handling Outliers

- Outliers are extreme values that differ significantly from other observations. They can distort statistical measures and model accuracy.

EDA and Machine Learning Notes

- Visual Checks: Boxplots highlight data spread and outliers using whiskers; scatter plots help identify unusual combinations of values.

- Statistical Detection: The IQR method flags values outside Q1 - 1.5*IQR and Q3 + 1.5*IQR. Z-score finds points more than 3 standard deviations from the mean.

- Treatment: Removing is safest when outliers are due to errors. Capping (winsorizing) keeps them within a reasonable range without losing data.

5. Exploratory Data Analysis (EDA)

- EDA is the step where you visually and statistically explore data to find trends, patterns, and relationships.

- Univariate Analysis: Looks at each feature separately to understand its distribution. Histograms and bar plots are common tools.

- Bivariate Analysis: Checks relationships between two variables — scatter plots reveal patterns; correlation heatmaps quantify relationships.

- Multivariate Analysis: Examines interactions between more than two variables. Pair plots are useful for continuous variables; grouped aggregations help summarize patterns.

6. Encoding Categorical Variables

- Machine learning models require numeric input, so categorical data must be encoded.

- Label Encoding: Assigns each category a numeric value. Works for ordinal variables (e.g., small=1, medium=2, large=3).

- One-Hot Encoding: Creates a separate binary column for each category, avoiding numeric relationships where none exist.

7. Feature Scaling

- Scaling ensures that features with different units or scales don’t dominate the model’s learning process.

- Standardization: Rescales data so it has mean 0 and standard deviation 1. Best for algorithms like SVM or logistic regression.

- Normalization: Rescales features to the range [0, 1], often useful for neural networks and KNN.

8. Feature Selection

EDA and Machine Learning Notes

- Feature selection removes irrelevant or redundant data, improving model accuracy and reducing computation.

- Correlation Check: High correlation (>0.85) between features can cause multicollinearity; remove one to simplify the model.

- Statistical Tests: Methods like chi-square test (for categorical) or ANOVA (for continuous) identify significant predictors.

- Recursive Feature Elimination: Automatically selects the most important features by testing combinations iteratively.

9. Splitting Data

- Splitting data into training and testing sets ensures the model’s performance is evaluated on unseen data.

- Common Split: 70–80% training, 20–30% testing. This balance provides enough data for training while keeping enough for testing.

- Function: Use train_test_split() in scikit-learn to randomly split data. Stratified splitting is recommended for imbalanced classification problems.

10. Saving Processed Data

- Once data is cleaned and processed, save it for future analysis or sharing.

- To CSV: Use to_csv('filename.csv', index=False) to avoid saving row indices.

- To Excel: Use to_excel('filename.xlsx', index=False) for Excel-friendly output.

- Machine Learning Algorithms & Use Cases (Detailed)

1. Lasso, Ridge, and Regression Case Studies

- Lasso Regression – Adds an L1 penalty to the loss function, forcing some coefficients to be exactly zero, thus performing feature selection. Useful in high-dimensional data.

- Ridge Regression – Adds an L2 penalty, shrinking coefficients but keeping all features. Helps with multicollinearity by distributing weight more evenly.

- Linear Regression – The simplest regression model assuming a linear relationship between input and output. Works best when residuals are normally distributed.

EDA and Machine Learning Notes

2. Logistic Regression, KNN, SVM

- Logistic Regression – Outputs probabilities for classes using a sigmoid function. Works for binary and multinomial classification tasks.

- KNN – Classifies by majority vote among nearest neighbors. Works well for non-linear boundaries but can be slow on large datasets.

- SVM – Finds the optimal separating hyperplane with maximum margin. Effective in high-dimensional spaces but can be sensitive to parameter choice.

3. Decision Tree and Random Forest

- Decision Tree – Splits data based on feature thresholds into simpler subsets. Easy to interpret but prone to overfitting.

- Random Forest – Combines many decision trees trained on bootstrapped datasets and random subsets of features, improving accuracy and reducing overfitting.

4. Bagging, Boosting, XGBoost, Voting

- Bagging – Builds multiple models independently on random samples and averages their predictions, reducing variance.

- Boosting – Builds models sequentially, with each model focusing on the errors of the previous one. Improves weak learners into strong ones.

- XGBoost – A highly optimized gradient boosting implementation with built-in regularization and parallel computation.

- Voting – Combines predictions from multiple models using majority rule (hard voting) or averaged probabilities (soft voting).

Cross-validation, ROC-AUC, GridSearchCV, Randomized Search CV

- Cross-Validation – Splits data into multiple folds to train and test on different subsets, giving a more reliable performance estimate. K-Fold divides into K equal folds; Stratified K-Fold maintains class balance.

- ROC-AUC – ROC curve plots TPR vs FPR for different thresholds. AUC measures the area under this curve, with 1 being perfect and 0.5 meaning random guessing.

- GridSearchCV – Tests all possible parameter combinations from a grid. Ensures best parameters are found but can be slow on large grids.

EDA and Machine Learning Notes

- Randomized Search CV – Tests a fixed number of random parameter combinations, much faster for large parameter spaces while still often finding near-optimal results.