Exploratory Data Analysis (EDA)

CMSC 173 - Machine Learning

Course Lecture

What is Exploratory Data Analysis?

Definition

EDA is the process of investigating datasets to summarize their main characteristics, often using statistical graphics and other data visualization methods.

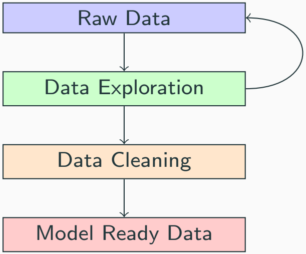
Primary Goals:

- **Understand** data structure and quality
- **Discover** patterns and relationships
- **Identify** anomalies and outliers
- **Guide** feature engineering decisions
- **Inform** modeling strategy

Key Questions EDA Answers:

- What does my data look like?
- Is my data clean and complete?
- What patterns exist?
- Which features are important?

EDA Process Overview:



Remember

EDA is iterative! Insights from one analysis often lead to new questions and deeper investigations.

Data Types in Tabular Data: Titanic Example

Example: Titanic Survival Data Set

Contains information on 1309 passengers aboard the Titanic and whether they survived or not. Goal: To predict the survival of passengers based on their attributes.

For tabular data, different data types can exist in one table.

ID	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	7.25	S
1	1	1	Cumings, Mrs. John Bradley	female	38.0	1	0	71.28	C
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	7.92	S
3	1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0	53.10	S
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	8.05	S

Integer, Ordinal — Integer, Binary — Integer, Categorical — String — String, Categorical — Continuous — Integer, Non-negative — Integer, Non-negative — Continuous — String, Categorical

Attributes:

- **Passenger ID** - An identifier unique to a passenger
- **Survived** - 1 = survived, 0 = did not survive
- **Pclass** - 1, 2, 3 = travel class
- **Name** - Passenger's name
- **Age** - Passenger's age
- **SibSp** - Number of siblings and spouses aboard
- **Parch** - Number of parents and children aboard
- **Ticket** - Ticket number
- **Fare** - Amount paid for ticket
- **Cabin** - Cabin of residence

Data Modalities and Types

Data Modalities:

- **Structured:** Tables, CSV, databases
- **Semi-structured:** JSON, XML, logs
- **Unstructured:** Text, images, audio, video

Structured Data Example

ID	Name	Age	Salary
1	Alice	25	50000
2	Bob	30	65000
3	Carol	28	58000

Data Attributes by Nature:

- **Quantitative:** Numerical measurements
- **Qualitative:** Categorical descriptions

Detailed Data Types:

Numerical (Quantitative)

- **Continuous:** Height, weight, temperature
- **Discrete:** Count of items, number of children

Categorical (Qualitative)

- **Nominal:** Colors, gender, country
- **Ordinal:** Grade (A,B,C), satisfaction levels
- **Binary:** Yes/No, True/False

Special Types:

- **DateTime:** Timestamps, dates
- **Text:** Free-form text, descriptions
- **Geospatial:** Coordinates, addresses
- **Mixed:** Combinations of above types

Data Understanding: First Look

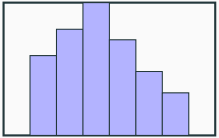
Initial Data Exploration:

- `df.shape` - Dimensions (rows, columns)
- `df.info()` - Data types and memory usage
- `df.head()`, `df.tail()` - Sample records
- `df.describe()` - Summary statistics
- `df.columns` - Column names

Titanic Dataset Example

```
df.shape: (891, 12)
df.info(): 714 non-null Age, 891 non-null Sex
Target: Survived (binary)
```

Quick Visual Overview:



Age Distribution

Key Insights from First Look:

- **Missing Data:** Age has 177 missing values
- **Data Types:** Mix of numerical and categorical
- **Target Variable:** Survived (binary classification)
- **Features:** 11 potential predictors

Handling Missing Values: Strategies

Missing Data Patterns:

- **MCAR:** Missing Completely At Random
- **MAR:** Missing At Random
- **MNAR:** Missing Not At Random

Detection Methods:

- `df.isnull().sum()`
- `df.info()`
- Heatmaps: `sns.heatmap(df.isnull())`
- Missing patterns visualization

Handling Strategies:

- Delete: Listwise/Pairwise deletion
- Impute: Mean, median, mode, KNN
- Model: Regression, ML-based imputation

Outlier Detection:

Statistical Methods

- **IQR Rule:** $Q_1 - 1.5 \times IQR$ to $Q_3 + 1.5 \times IQR$
- **Z-score:** $|z| > 3$ (or 2.5)
- **Modified Z-score:** Using median

Visual Methods

- Box plots, violin plots
- Scatter plots for bivariate
- Histograms with overlays

Advanced Methods

- Isolation Forest
- Local Outlier Factor (LOF)
- DBSCAN clustering

Column Transformations

Common Transformations:

- **Encoding:** Convert categories to numbers
- **Scaling:** Normalize numerical ranges
- **Binning:** Group continuous values
- **Feature Creation:** Derive new features

Categorical Encoding

One-Hot Encoding:

Color	Red	Blue	Green
Red	1	0	0
Blue	0	1	0
Green	0	0	1

Label Encoding:

Grade	Encoded
A	3
B	2
C	1
F	0

Feature Engineering Examples:

From DateTime

```
df['hour'] = df['timestamp'].dt.hour
df['is_weekend'] = df['day_of_week'] > 4
```

From Text

```
df['text_length'] = df['review'].str.len()
df['has_exclamation'] = df['review'].str.contains('!')
```

Mathematical Transformations

```
df['log_income'] = np.log1p(df['income'])
df['age_income'] = df['age'] * df['income']
```

Data Normalization and Scaling

Why Normalize?

- Different scales affect ML algorithms
- Features with larger ranges dominate
- Improves convergence speed
- Required for distance-based algorithms

Min-Max Scaling

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

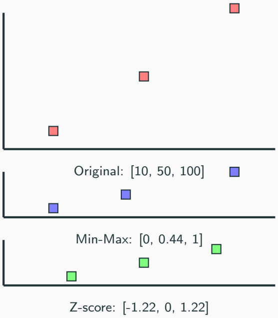
Range: [0, 1] **Use:** When you know min/max bounds

Z-Score Standardization

$$X_{scaled} = \frac{X - \mu}{\sigma}$$

Range: $\mu = 0, \sigma = 1$ **Use:** When data is normally distributed

Scaling Methods Comparison:



Python Implementation

```
scaler = MinMaxScaler()
X_minmax = scaler.fit_transform(X)
scaler = StandardScaler()
X_standard = scaler.fit_transform(X)
```

Distribution Analysis and Visualization Examples

Numerical Variables:

- **Central Tendency:** Mean, median, mode
- **Spread:** Variance, std dev, range, IQR
- **Shape:** Skewness, kurtosis
- **Normality Tests:** Shapiro-Wilk, Anderson-Darling

Skewness Interpretation

- $-0.5 < \text{Skew} < 0.5$: Symmetric
- $0.5 < |\text{Skew}| < 1$: Moderate
- $|\text{Skew}| > 1$: Highly skewed

Transformations:

- **Log:** Right-skewed data
- **Square root:** Moderate skew
- **Box-Cox:** Optimal transformation

Categorical Variables:

- **Frequency tables:** Value counts
- **Proportions:** Relative frequencies
- **Mode:** Most frequent category
- **Cardinality:** Number of unique values

Key Considerations

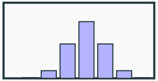
- High cardinality $\rightarrow$ Dimensionality issues
- Rare categories $\rightarrow$ Grouping strategies
- Ordinal vs nominal encoding

Visualization Tools:

- Bar charts: Category frequencies
- Pie charts: Proportions (use sparingly)
- Count plots: Seaborn implementation
- Word clouds: Text data overview

Visualization Gallery: Common EDA Plots

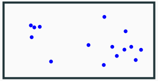
Univariate Visualizations:



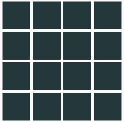
Histogram

Box Plot

Bivariate Visualizations:

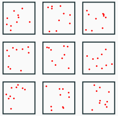


Scatter Plot



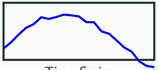
Correlation Heatmap

Multivariate Visualizations:



Pair Plot

Time Series:



Time Series

Python Code Examples

```
sns.histplot(df['age'])
sns.scatterplot(x='age', y='salary', data=df)
sns.pairplot(df)
sns.heatmap(df.corr(), annot=True)
```

Feature Selection: Choosing the Right Variables

Why Feature Selection?

- **Curse of dimensionality** - Too many features
- **Overfitting** - Model memorizes noise
- **Training time** - Computational efficiency
- **Interpretability** - Simpler models

Selection Methods:

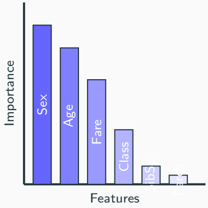
- **Filter Methods:** Statistical tests, correlation
- **Wrapper Methods:** Forward/backward selection
- **Embedded Methods:** LASSO, Random Forest importance

Correlation-based Selection

```
corr_matrix = df.corr().abs()
to_drop = [col for col in upper_tri.columns

phantom
ldots
ldots
ldots
ldotsif any(upper_tri[col] > 0.95)]
```

Feature Importance Visualization:



Sklearn Feature Selection

```
from sklearn.feature_selection import SelectKBest
selector = SelectKBest(f_classif, k=5)
X_selected = selector.fit_transform(X, y)

rf = RandomForestClassifier()
importances = rf.feature_importances_
```

Correlation and Advanced Relationships

Correlation Types:

- **Pearson:** Linear relationships (continuous)
- **Spearman:** Monotonic relationships (ordinal)
- **Kendall:** Rank-based, robust to outliers

Interpretation

- $|r| < 0.3$: Weak
- $0.3 \leq |r| < 0.7$: Moderate
- $|r| \geq 0.7$: Strong
- $r = 0$: No linear correlation

Multicollinearity:

- **VIF:** Variance Inflation Factor
- **Condition Index:** Eigenvalue analysis
- **Tolerance:** $1 - R^2$ from regression

Visualization Techniques:

- Heatmaps: Correlation matrices
- Scatter plots: Pairwise relationships
- Pair plots: Multiple variables
- Parallel coordinates: High-dimensional

Advanced Analysis

Categorical-Numerical:

- ANOVA F-test
- Kruskal-Wallis test
- Box plots by category

Categorical-Categorical:

- Chi-square test
- Cramér's V
- Contingency tables

Feature Selection Example: Titanic Dataset

Available Features:

- Pclass - Passenger class (1,2,3)
- Sex - Gender (male/female)
- Age - Age in years
- SibSp - Siblings/spouses aboard
- Parch - Parents/children aboard
- Fare - Ticket fare
- Embarked - Port of embarkation
- Name - Passenger name
- Ticket - Ticket number
- Cabin - Cabin number

Feature Importance (Random Forest)

Feature	Importance
Sex	0.31
Age	0.28
Fare	0.23
Pclass	0.12
SibSp	0.04
Parch	0.02

Selection Strategy:

Remove

- Name - High cardinality, no pattern
- Ticket - Unique identifiers
- Cabin - 77% missing

Engineer

- Title - Extract from Name (Mr, Mrs, Dr)
- FamilySize - SibSp + Parch + 1
- IsAlone - FamilySize == 1
- AgeBin - Bin Age into groups

Final Feature Set:

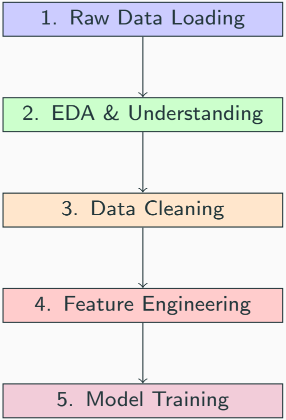
- Pclass, Sex, Age, Fare
- Embarked, Title, FamilySize, IsAlone

Performance Impact:

Feature Set	Accuracy
All original	0.79
Selected + Engineered	0.83

Complete ML Pipeline: From Raw Data to Model

Pipeline Stages:



Sample Pipeline Code

```
pipeline = Pipeline([

phantom
ldots('scaler', StandardScaler()),

phantom
ldots('classifier', RandomForestClassifier())
texttt])
pipeline.fit(X_train, y_train)
accuracy = pipeline.score(X_test, y_test)
```

Key Benefits:

- Reproducible experiments
- Prevents data leakage
- Easy hyperparameter tuning
- Simplified deployment

Visualization Best Practices

Chart Selection Guide:

- **Distribution:** Histogram, KDE, box plot
- **Comparison:** Bar chart, grouped bar
- **Relationship:** Scatter plot, line plot
- **Composition:** Stacked bar, pie (avoid)
- **Trend:** Line plot, area chart

Design Principles

- Clarity: Simple, uncluttered
- Accuracy: Proper scaling, no misleading
- Efficiency: Maximize data-ink ratio
- Aesthetics: Professional appearance

Common Pitfalls:

- × Truncated y-axes
- × 3D charts without purpose
- × Too many colors/categories
- × Poor aspect ratios
- × Missing context/labels

Python Libraries:

- **Matplotlib:** Low-level, flexible
- **Seaborn:** Statistical plots, attractive defaults
- **Plotly:** Interactive, web-ready
- **Bokeh:** Interactive, large datasets
- **Altair:** Grammar of graphics

Key Functions:

- `sns.pairplot()`, `sns.heatmap()`
- `plt.subplots()`, `plt.hist()`
- `df.plot()`, `df.hist()`

EDA Workflow and Tools

Systematic EDA Process:

1. **Initial Assessment**
   - `df.shape`, `df.info()`
   - `df.describe()`
   - `df.head()`, `df.tail()`
2. **Data Quality Check**
   - Missing values analysis
   - Duplicate detection
   - Data type validation
3. **Univariate Analysis**
   - Distributions, outliers
   - Summary statistics
4. **Bivariate/Multivariate**
   - Correlations, relationships
   - Feature interactions

Automated EDA Tools:

- **Pandas Profiling:** Comprehensive reports
- **AutoViz:** Automatic visualization
- **SweetViz:** Interactive HTML reports
- **DataPrep:** Fast EDA with minimal code

Key Metrics to Track

- **Completeness:** % non-missing values
- **Uniqueness:** Duplicate rate
- **Validity:** Data type consistency
- **Distribution:** Skewness, normality
- **Outliers:** Percentage beyond thresholds

Documentation:

- Data dictionary
- Assumptions and decisions
- Transformation rationale
- Quality issues found

Summary and Next Steps

EDA Deliverables:

- **Data Profile:** Shape, types, quality
- **Quality Report:** Missing, outliers, issues
- **Statistical Summary:** Distributions, correlations
- **Visualizations:** Key patterns and relationships
- **Insights:** Business-relevant findings

Critical Questions Answered

- What is the data quality?
- What patterns exist?
- Which features are important?
- What preprocessing is needed?
- Are there data collection issues?

Post-EDA Actions:

1. **Data Cleaning**
   - Handle missing values
   - Remove/treat outliers
   - Fix data quality issues
2. **Feature Engineering**
   - Create new features
   - Transform distributions
   - Encode categorical variables
3. **Feature Selection**
   - Remove redundant features
   - Select informative variables
   - Address multicollinearity
4. **Model Preparation**
   - Split data
   - Scale/normalize
   - Validation strategy

Remember: EDA is iterative and informs all subsequent ML pipeline decisions.