Lecture 8: Data Exploration (EDA) Lecture Notes

Dr. Ratnesh Srivastava, CSIT, GGV Bilaspur

July 22, 2025

1 Introduction to Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a crucial step in the data science pipeline. It involves analyzing data to summarize their main characteristics, often with visual methods. EDA helps in understanding the dataset, uncovering patterns, detecting anomalies, testing hypotheses, and checking assumptions with the help of summary statistics and graphical representations.

2 Key Tools for EDA

Several powerful Python libraries facilitate effective EDA:

- **Matplotlib:** A comprehensive library for creating static, animated, and interactive visualizations in Python. It provides fine-grained control over plots.
- **Seaborn:** Built on top of Matplotlib, Seaborn provides a high-level interface for drawing attractive and informative statistical graphics. It simplifies the creation of complex visualizations like heatmaps and pairplots.
- **Plotly:** An interactive graphing library that makes it easy to create publication-quality plots. Plotly supports various plot types, including 3D plots, and offers interactive features for web-based dashboards.
- **Pandas-profiling:** This library generates interactive HTML reports for pandas DataFrames. It quickly performs a comprehensive EDA, including statistical summaries, missing value analysis, and correlation analysis, with just a few lines of code.

3 Important Steps in EDA

3.1 1. Initial Data Understanding

Before diving deep, it's essential to get a basic understanding of your data.

- **Check Data Types (`df.info()`, `df.dtypes`):** Understand the data type of each column (e.g., numerical, categorical, datetime).
- **Handle Null Values (`df.isnull().sum()`):** Identify missing data and decide on strategies for handling them (e.g., imputation, removal).
- **Descriptive Statistics (`df.describe()`):** Obtain statistical summaries for numerical columns (mean, median, standard deviation, quartiles, etc.).
  - **Mathematical Intuition:**
    - **Mean (μ or $\bar{x}$):** The sum of all values divided by the number of values. It's the "average" and represents the central tendency.

      $$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

      Intuition: If you were to balance a seesaw with data points, the mean is where the fulcrum would be placed. It's sensitive to extreme values (outliers).

  * **Median:** The middle value in a dataset when values are ordered. If there's an even number of observations, it's the average of the two middle values. *Intuition:* The median divides the data into two equal halves. It's robust to outliers.
  * **Standard Deviation (σ or s):** Measures the typical distance between data points and the mean. A small standard deviation indicates data points are close to the mean; a large one indicates data points are spread out.

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

    Intuition: It quantifies the "spread" or "dispersion" of your data.
  * **Quartiles (Q1, Q2, Q3):** Divide the ordered data into four equal parts.
    · Q1 (25th percentile): 25% of data falls below this value.
    · Q2 (50th percentile): This is the Median.
    · Q3 (75th percentile): 75% of data falls below this value.

    Intuition: They help us understand the distribution across different segments of the data, not just the center.

- **Check for Duplicates (`df.duplicated().sum()`):** Identify and handle duplicate rows, as they can bias analysis.

```
import pandas as pd
import numpy as np

# Sample Data
data = {
    'Age': [25, 30, 35, 40, 28, np.nan, 32, 45, 30, 25],
    'Salary': [50000, 60000, 75000, 90000, 55000, 70000, 62000, 95000, 60000, 52000],
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Female'],
    'Experience': [2, 5, 8, 10, 3, 6, 4, 12, 5, 2]
}
df = pd.DataFrame(data)

print("--- DataFrame Info ---")
df.info()

print("\n--- Null Values ---")
print(df.isnull().sum())

print("\n--- Descriptive Statistics for Age and Salary ---")
print(df[['Age', 'Salary']].describe())

print("\n--- Duplicate Rows ---")
print(df.duplicated().sum())
```

Example Walkthrough: From `df[['Age', 'Salary']].describe()`:

- **Age:** The mean age is approximately 32.78 years, but the median is 30. This difference suggests a slight skew or perhaps an outlier pulling the mean up. The standard deviation is about 6.5 years, indicating that ages typically vary by 6.5 years from the mean.
- **Salary:** The mean salary is around $66,900, and the median is $61,000. This again suggests a slight skew towards higher salaries. The standard deviation of approximately $14,460 shows the typical spread of salaries.

3.2 2. Distribution Analysis

Understanding the distribution of individual variables is fundamental.

- **Histograms:** Visualize the distribution of numerical data. They show the frequency of data points within different bins.
  - **Mathematical Intuition:** Histograms approximate the Probability Density Function (PDF) for continuous data or Probability Mass Function (PMF) for discrete data. Each bar's height represents the frequency or count of observations falling into a specific range (bin).
- **Kernel Density Estimates (KDEs):** A smoothed version of a histogram, showing the probability density function of a continuous variable.
  - **Mathematical Intuition:** KDEs use a "kernel" (a smoothing function, often a Gaussian or normal distribution) placed at each data point. These kernels are then summed to estimate the overall density curve. It provides a non-parametric way to estimate the PDF.

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
sns.histplot(df['Age'].dropna(), kde=True, bins=5)
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')

plt.subplot(1, 2, 2)
sns.histplot(df['Salary'], kde=True, bins=5)
plt.title('Distribution of Salary')
plt.xlabel('Salary')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

3.3 3. Boxplots and Violin Plots

These plots are excellent for visualizing the distribution and spread of numerical data, especially across different categories, and for identifying potential outliers.

- **Boxplots:** Display the five-number summary (minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum) of a set of data. Outliers are often plotted as individual points.
  - **Mathematical Intuition:** The box represents the Interquartile Range (IQR) = Q3 - Q1, which contains the middle 50% of the data. The median splits this box. The "whiskers" extend to the minimum and maximum values within 1.5 * IQR of Q1 and Q3, respectively. Points beyond the whiskers are considered potential outliers.
- **Violin Plots:** Combine the features of a boxplot and a KDE plot. They show the probability density of the data at different values (like a KDE) along with the quartiles (like a boxplot).
  - **Mathematical Intuition:** The width of the "violin" at any given point represents the density of data points at that value. A wider section means more data points are concentrated there. It provides a visual representation of the underlying data distribution's shape.

```
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
sns.boxplot(x='Gender', y='Salary', data=df)
plt.title('Salary Distribution by Gender (Boxplot)')
plt.subplot(1, 2, 2)
sns.violinplot(x='Gender', y='Age', data=df)
plt.title('Age Distribution by Gender (Violin Plot)')

plt.tight_layout()
plt.show()
```

3.4 4. Correlation Heatmaps

Correlation heatmaps are used to visualize the correlation matrix of numerical variables. This helps in understanding the linear relationship between different features.

- **Mathematical Intuition (Pearson Correlation Coefficient - r):** The most common type of correlation measures the strength and direction of a linear relationship between two continuous variables. Its value ranges from -1 to +1.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

  - $r = 1$: Perfect positive linear relationship (as one variable increases, the other increases proportionally).
  - $r = -1$: Perfect negative linear relationship (as one variable increases, the other decreases proportionally).
  - $r = 0$: No linear relationship (variables are independent in a linear sense).

  Intuition: It tells you how closely two variables "move together." If `Age` and `Experience` have a high positive correlation (e.g., 0.9), it means that as a person's age increases, their experience tends to increase significantly.

```
plt.figure(figsize=(8, 6))
correlation_matrix = df[['Age', 'Salary', 'Experience']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Features')
plt.show()
```

Example Walkthrough:

- You might observe a strong positive correlation between 'Age' and 'Experience' (e.g., 0.95). This makes intuitive sense: older people typically have more work experience.
- You might also see a positive correlation between 'Experience' and 'Salary' (e.g., 0.8). This indicates that generally, more experienced individuals tend to earn higher salaries.
- A lower correlation between 'Age' and 'Salary' (e.g., 0.7) compared to 'Experience' and 'Salary' might suggest that while age plays a role, experience is a more direct driver of salary.

3.5 5. Pairplots

Pairplots (also known as scatterplot matrices) are used to visualize the relationships between multiple variables. They create a grid of scatterplots for each pair of variables and histograms for individual variables.

- **Mathematical Intuition:** Each scatterplot in a pairplot visually represents the relationship between two variables. If data points tend to form a line, there's a linear correlation. If they form a curve, there's a non-linear relationship. If they are scattered randomly, there's little to no relationship. Histograms/KDEs on the diagonal provide the univariate distribution for each variable, as discussed before.

```
sns.pairplot(df[['Age', 'Salary', 'Experience', 'Gender']], hue='Gender', diag_kind='kde')
plt.suptitle('Pairplot of Numerical Features by Gender', y=1.02)
plt.show()
```

3.6 6. Outlier Detection

Outliers are data points that significantly differ from other observations. Identifying and handling outliers is crucial as they can skew analysis results.

- **Boxplots:** Visually identify outliers as points outside the "whiskers."
- **Z-score:** Measures how many standard deviations away a data point is from the mean.

  $$Z = \frac{x - \mu}{\sigma}$$

  Intuition: A data point with a high absolute Z-score (e.g., $|Z| > 3$) is considered an outlier because it's many standard deviations away from the mean, making it an unlikely observation if the data were normally distributed.
- **IQR (Interquartile Range) Method:** Defines outliers as points beyond 1.5 times the IQR from the first or third quartile.
  - **Lower Bound:** $Q1 - 1.5 \times IQR$
  - **Upper Bound:** $Q3 + 1.5 \times IQR$

  Intuition: This method defines a "normal" range for the middle 50% of the data and extends it by a certain factor (1.5). Any data point falling outside this extended range is considered an outlier. It's less sensitive to extreme values than the Z-score method because it uses medians and quartiles instead of the mean and standard deviation.

```
Q1_salary = df['Salary'].quantile(0.25)
Q3_salary = df['Salary'].quantile(0.75)
IQR_salary = Q3_salary - Q1_salary

outlier_threshold_upper = Q3_salary + 1.5 * IQR_salary
outlier_threshold_lower = Q1_salary - 1.5 * IQR_salary

outliers = df[(df['Salary'] < outlier_threshold_lower) | (df['Salary'] > outlier_threshold_upper)]

print("\n--- Outliers in Salary (IQR Method) ---")
print(outliers)
```

Example Walkthrough: Let's assume for our sample `Salary` data:

- Sorted `Salary`: [50000, 52000, 55000, 60000, 60000, 62000, 70000, 75000, 90000, 95000]
- Q1 (25th percentile) = \$55000 (average of 3rd and 4th values: 52000, 55000, 60000; or more precisely, it's the 0.25*(n+1)th value if considering sorted array, here 0.25 * 10 = 2.5th value, so between 52000 and 55000). For pandas quantile it would be 55000 based on interpolation.
- Q3 (75th percentile) = \$75000
- IQR = \$75000 - \$55000 = \$20000
- Lower Threshold = \$55000 - 1.5 * \$20000 = \$55000 - \$30000 = \$25000
- Upper Threshold = \$75000 + 1.5 * \$20000 = \$75000 + \$30000 = \$105000

Any salary below \$25000 or above \$105000 would be considered an outlier by this method. In our sample data, there are no outliers based on this calculation.

3.7 7. Automated EDA with Pandas-profiling

`pandas-profiling` automates much of the EDA process, generating a comprehensive report with a single line of code.

```
# !pip install pandas-profiling  # Uncomment to install if not already installed
from ydata_profiling import ProfileReport

profile = ProfileReport(df, title="EDA Report for Sample Data")
# profile.to_file("sample_eda_report.html") # Uncomment to save the report to an HTML file
print("\n--- Pandas-profiling Report Generated ---")
# The report can be viewed in a Jupyter Notebook directly or saved as an HTML file.
```

4 Advanced EDA Concepts (from Article)

The article "Mastering Exploratory Data Analysis (EDA): Everything You Need To Know" by Sze Zhong LIM on Data And Beyond highlights a three-level approach to EDA:

- **EDA Level 0 — Pure Understanding of Original Data:** Focuses on initial checks like `df.head()`, `df.describe()`, `df.duplicated().sum()`, and basic histograms for numerical columns to get a preliminary grasp of the dataset.
- **EDA Level 1 — Transformation of Original Data:** Involves data cleaning and preparation steps such as standardizing column names, filling null values, changing data types, data validation, and mapping/binning categorical features. This level emphasizes data quality and usability.
- **EDA Level 2 — Understanding of Transformed Data:** Delves into deeper analysis using transformed data. This includes:
  - **Correlation Analysis:** As discussed, identifying relationships between features.
  - **Information Value (IV) and Weight of Evidence (WOE):** Techniques primarily used in credit scoring and risk modeling to understand the predictive power of a feature with respect to a target variable.
    * **Mathematical Intuition (WOE and IV):**
      - **Weight of Evidence (WOE):** Measures the strength of a categorical independent variable's relationship to a binary dependent variable (e.g., default/non-default). For a given category (bin) i:

$$WOE_i = \ln\left(\frac{\text{\% of Non-Events in Bin } i}{\text{\% of Events in Bin } i}\right)$$

        Intuition: If `WOE` is positive, that category is associated with more "non-events" (good outcomes); if negative, it's associated with more "events" (bad outcomes). The larger the absolute `WOE`, the stronger the relationship.
      - **Information Value (IV):** Summarizes the predictive power of a feature. It's the sum of `WOE` multiplied by the difference in event/non-event proportions for each bin.

$$IV = \sum_{i=1}^{n}(\text{\% of Non-Events in Bin } i - \text{\% of Events in Bin } i) \times WOE_i$$

        Intuition: A higher `IV` value indicates that the variable is a better predictor of the target variable. Common thresholds for IV interpretation exist (e.g., <0.02: useless, 0.02-0.1: weak, 0.1-0.3: medium, >0.3: strong).
  - **Feature Importance from ML Models:** Deriving insights into which features are most influential in predicting the target variable from various machine learning models (e.g., Decision Tree, Random Forest, XGBoost, Logistic Regression).

* **Mathematical Intuition:** Different models calculate feature importance differently. For tree-based models (like Random Forests), importance is often based on how much each feature reduces impurity (e.g., Gini impurity or entropy) across all the trees in the forest. Features that lead to larger reductions in impurity when splitting are considered more important. For linear models (like Logistic Regression), importance can be inferred from the absolute value of the coefficients (after standardization), where larger absolute coefficients indicate a stronger impact on the predicted outcome.

5 Questions and Answers

Q1: Why is EDA considered a crucial step in the data science workflow? A1: EDA helps data scientists gain insights into the dataset's structure, identify patterns, detect anomalies, test hypotheses, and validate assumptions. It guides subsequent steps like feature engineering, model selection, and hyperparameter tuning, leading to more robust and accurate models. Without EDA, one might build a model on flawed data or miss crucial relationships, leading to incorrect conclusions or poor model performance.

Q2: What are the main differences between a boxplot and a violin plot? When would you choose one over the other? A2: A **boxplot** primarily displays the five-number summary (minimum, Q1, median, Q3, maximum) and highlights individual outliers. Its strength is in clearly showing the central tendency, spread, and the presence of extreme values. A **violin plot**, while also showing the five-number summary (often through a small box inside), additionally displays the probability density of the data at different values using a smoothed kernel density estimate. This provides a richer view of the data's distribution shape, revealing multimodality or skewness that a boxplot might hide. You would choose a **boxplot** when you need a concise summary of spread and clear outlier identification. You would choose a **violin plot** when you need to understand the full shape of the distribution, especially when comparing distributions across different categories, as it can reveal nuances like multiple peaks in the data.

Q3: Explain the significance of correlation heatmaps in EDA, specifically referencing the Pearson correlation coefficient. A3: Correlation heatmaps visually represent the Pearson correlation coefficient (r) between numerical variables. The Pearson correlation coefficient measures the strength and direction of a *linear* relationship between two variables, ranging from -1 to +1. A value close to +1 indicates a strong positive linear relationship (as one variable increases, the other tends to increase). A value close to -1 indicates a strong negative linear relationship (as one variable increases, the other tends to decrease). A value close to 0 suggests no linear relationship. The significance lies in:

1. **Identifying highly related features:** This can be useful for feature selection (e.g., if two features are highly correlated, one might be redundant).
2. **Detecting multicollinearity:** High correlations between independent variables can cause issues in certain statistical models (e.g., linear regression), leading to unstable coefficient estimates.
3. **Understanding trends:** It quickly shows which variables tend to move together, or inversely, in the dataset.

Q4: How can `pandas-profiling` assist in accelerating the EDA process? A4: `pandas-profiling` automates the generation of a detailed, interactive EDA report with a single line of code. This report comprehensively covers:

- **Statistical summaries:** Mean, median, standard deviation, quartiles, etc., for numerical features.
- **Missing value analysis:** Counts and percentages of missing values.
- **Correlation matrices:** Various types of correlation (Pearson, Spearman, Kendall, Phik) visually represented.
- **Distribution plots:** Histograms for numerical data, bar charts for categorical data.
- **Categorical variable details:** Frequencies, unique values.

This automation saves significant time and effort that would otherwise be spent manually coding each EDA step, especially for large datasets with many variables. It provides a quick and thorough overview, allowing data scientists to focus on deeper insights rather than repetitive tasks.

Q5: What are outliers, and why is their detection important in EDA? Provide the mathematical formulation for the IQR method of outlier detection and explain its intuition.
A5: Outliers are data points that significantly deviate from the general pattern or distribution of the majority of the data. Their detection is crucial in EDA because:

1. **Impact on Statistics:** Outliers can heavily skew descriptive statistics like the mean and standard deviation, leading to misinterpretations of the data's central tendency and spread.
2. **Model Performance:** In machine learning, outliers can adversely affect model training, causing models to perform poorly or generalize incorrectly to new data. They can introduce noise and lead to biased parameter estimates.
3. **Anomalies:** Sometimes, outliers represent genuine anomalies or errors in data collection (e.g., a typo in a measurement), which need to be investigated and possibly corrected.

IQR (Interquartile Range) Method for Outlier Detection: This method defines a range within which "normal" data points are expected to fall.

1. Calculate the First Quartile ($Q1$): The 25th percentile of the data.
2. Calculate the Third Quartile ($Q3$): The 75th percentile of the data.
3. Calculate the Interquartile Range (IQR):
$$IQR = Q3 - Q1$$
4. Define the outlier boundaries:
   - **Lower Bound:** $Q1 - 1.5 \times IQR$
   - **Upper Bound:** $Q3 + 1.5 \times IQR$

Any data point (x_i) such that $x_i <$ Lower Bound or $x_i >$ Upper Bound is considered an outlier.

Mathematical Intuition: The IQR represents the spread of the middle 50% of the data. By extending this range by a factor of 1.5 in both directions ($1.5 \times IQR$), we create a "fence." Data points that fall outside this fence are considered statistically unusual. This method is robust to extreme values because it relies on quartiles (which are median-based) rather than the mean and standard deviation, making it suitable for non-normally distributed data.

References:

- Mastering Exploratory Data Analysis (EDA): Everything You Need To Know - Sze Zhong LIM, Data And Beyond