AIG150- Week 3

Exploratory Data Analysis

Reading Text:

Ch 07,10,11,12: Pandas for everyone

Ch 13: Python for Data Science for Dummies

Agenda

Data Munging

- Pandas Data types
- Type Conversions
- String Processing

Dates and Times

Exploratory Data Analysis

- Defining descriptive Statistics for Numeric Data
- Counting for Categorical Data
- Creating Applied Visualization for EDA Understanding Correlation
- Modifying Data Distributions

Pandas Data Types

- integers (Int64)
- float (float64) object → Strings
- category
- Boolean (bool)
- Datetime64
- Timedelta[ns]

Type Conversions

- dtype() prints the data types of a variable
- Series.astype()
- Ex: to convert a col to a string → `data['col']=data['col'].astype(str)`
- You can also check the to_numeric() function for numeric conversions, check the documentation for "errors" parameter
- Category data type is memory and speed efficient and have specialized functions to work with, check the documentation for categorial accessor

Exploratory Data Analysis

- Load online data: Choose a suitable online dataset and load it into a pandas DataFrame.
- Inspect data types: Examine the data types of the columns in the loaded DataFrame.
- Analyze data distribution: Generate visualizations or statistics to understand the distribution of data in relevant columns (e.g., numerical and categorical columns).
- Summarize findings: Provide a brief summary of the data types and distributions observed.
- Finish task: Put the findings from the earlier stages into a format that anyone can read.

Exploratory Data Analysis

- A general approach to explore datasets by using simple summary statistics and graph visualization
- We need to make sure data is worthy to work with → garbage in/garbage out
- Describe the data
- Closely explore data distributions Understand the relations between variables
- Notice unusual or unexpected situations. Place the data into groups
- Notice unexpected patterns within groups
- Take note of group differences

Descriptive Statistics

- Count Mean
- Standard deviation
- Minimum value 25th percentile
- 50th percentile (median)
- 75th percentile
- Maximum value

Frequency Distribution Analysis for Categorical Columns:

Descriptive Statistics for Numerical Columns in df_housing:

```
df_housing.describe()
```

```
categorical_cols =
df_housing.select_dtypes(include='object').columns
value_counts = df_housing[col].value_counts()
```

Visualization for EDA

- Bivariate :see relations in couple of variables
- Multivariate: Considering more than two variables at the same time Boxplots
- Scatter plots Heatmaps
- Histograms

Visualization plots

Histplot (Histogram):

Purpose: To show the distribution of a single numerical variable.

What it shows: The frequency (or count) of data points within different bins or intervals of the variable's range. It helps you see the shape of the distribution (e.g., normal, skewed, bimodal), where the data is centered, and how spread out it is.

Scatter plot:

Purpose: To show the relationship between two numerical variables.

What it shows: Each point on the plot represents an observation, with its position determined by the values of the two variables. Scatter plots help visualize patterns, trends (e.g., linear, non-linear), clusters, and potential outliers in the relationship between two variables.

Boxplot (Box and Whisker Plot):

Purpose: To show the distribution and summary statistics of a numerical variable, often grouped by a categorical variable.

What it shows: A boxplot displays the median, quartiles (25th and 75th percentiles), and potential outliers of a dataset. The box represents the interquartile range (IQR), the line inside the box is the median, and the "whiskers" extend to show the range of the data, excluding outliers. It's excellent for comparing the distribution of a numerical variable across different categories.

Visualization plots

Pairplot:

Purpose: To visualize the pairwise relationships between multiple variables in a dataset.
What it shows: A grid of plots where each variable is plotted against every other variable. On the diagonal, it typically shows the distribution of each individual variable (often a histogram or KDE plot). The off-diagonal plots are scatter plots showing the relationship between each pair of variables. It's a quick way to get an overview of the relationships and distributions within your data.

Heatmap:

Purpose: To visualize the correlation or relationship matrix between numerical variables.
What it shows: A grid where each cell represents the strength and direction of the relationship (usually correlation) between two variables. The color intensity and shade of each cell correspond to the value of the correlation coefficient. It's a good way to quickly identify which pairs of variables have strong positive or negative correlations.

Matplotlib and Seaborn

import matplotlib.pyplot as plt:

matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
matplotlib.pyplot is a module within Matplotlib that provides a convenient, stateful interface for plotting. It's designed to work much like MATLAB.
as plt is a standard convention to give the matplotlib.pyplot module a shorter alias, plt. This allows you to call functions from this module using plt. followed by the function name (e.g., plt.plot(), plt.figure(), plt.show()).

import seaborn as sns:

seaborn is a Python data visualization library based on Matplotlib.
It provides a high-level interface for drawing attractive and informative statistical graphics.
Seaborn is particularly good at visualizing relationships between variables and distributions.
as sns is the standard convention to give the Seaborn library the alias sns, so you can call its functions using sns. (e.g., sns.histplot(), sns.boxplot(), sns.heatmap()).

Visualization plots in Python

```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_housing, x='area', y='price')
plt.title('Price vs Area’)
plt.xlabel('Area’)
plt.ylabel('Price’)
plt.show()
```

```
sns.boxplot(data=df_housing, x='furnishingstatus', y='price’)
sns.pairplot(df_housing[numerical_subset])
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f")
sns.histplot(data=df_housing, x=col, kde=True)
```

Understanding Frequencies

- The mode of the frequency distribution that is the most frequent category
- The other most frequent categories, especially when they are comparable with the mode (bimodal distribution) or if there is a large difference between them
- The distribution of frequencies among categories, if rapidly decreasing or equally distributed Rare categories

Understanding Correlation

- Working with numeric variables, the estimate is a correlation, and the Pearson’s correlation is the most famous one. The Pearson’s correlation is the foundation for complex linear estimation models

- If the single values of two variables are usually above or below their respective averages, the two variables have a positive association, it means that they tend to agree, and you can figure out the behavior of one of the two by looking at the other. In such a case, their covariance will be a positive number, and the higher the number, the higher the agreement.

- If one variable is usually above and the other variable usually below their respective averages, the two variables are negatively associated. Even though the two disagree, it’s an interesting situation for making predictions, because by observing the state of one of them, you can figure out the likely state of the other (albeit they’re opposite). In this case, their covariance will be a negative number.

- A third state is that the two variables don’t systematically agree or disagree with each other. In this case, the covariance will tend to be zero, a sign that the variables don’t share much and have independent behaviors.

Understanding Correlation

- Correlations can work fine when your variables are numeric, and their relationship is strictly linear.

- For ordinal feature (a numeric variable but with orderings) or if you have nonlinearity due to non-normal distributions in your data, you can use nonparametric correlation, such as a Spearman.

- A *Spearman correlation* transforms your numeric values into rankings and then correlates the rankings, thus minimizing the influence of any nonlinear relationship between the two variables under scrutiny. The resulting correlation, commonly denoted as *rho,* is to be interpreted in the same way as a Pearson's correlation.

- When you work with categorical variables, the estimate is an association, and the chi-square statistic is the most frequently used tool for measuring association between features. Chí-square test is another nonparametric test for relationship when working with cross tables, works with both numeric and categorical data.

Measure of Normality In Data

Skewness

Kurtosis

Skewness:

What it measures: Skewness measures the asymmetry of the probability distribution of a real-valued random variable about its mean. In simpler terms, it tells you if the distribution is symmetrical or if it's "pushed" or "pulled" to one side.

Interpretation:

- **Zero Skewness:** Indicates a perfectly symmetrical distribution, like a normal distribution (bell curve). The data is evenly distributed around the mean.
- **Positive Skewness (Right Skew):** The tail on the right side of the distribution is longer or fatter than the tail on the left side. This means there are more data points on the left side of the mean, and a few larger values pull the mean to the right of the median.
- **Negative Skewness (Left Skew):** The tail on the left side of the distribution is longer or fatter than the tail on the right side. This means there are more data points on the right side of the mean, and a few smaller values pull the mean to the left of the median.

Kurtosis:

What it measures: Kurtosis measures the "tailedness" and "peakedness" of a distribution relative to a normal distribution. It describes the shape of the tails and how sharp or flat the peak is.

Interpretation (often relative to a normal distribution which has a Kurtosis of 0 in some definitions, or 3 in others - pandas uses a definition where normal distribution kurtosis is 0):

- **Mesokurtic (Kurtosis ≈ 0):** The distribution has a peak and tail thickness similar to a normal distribution.
- **Leptokurtic (Kurtosis > 0):** The distribution has a sharper, taller peak and fatter, heavier tails than a normal distribution. This means there are more outliers (extreme values) than in a normal distribution.
- **Platykurtic (Kurtosis < 0):** The distribution has a flatter, broader peak and thinner, lighter tails than a normal distribution. This means there are fewer outliers than in a normal distribution.

```
A perfect Gaussian distribution has Skewness = 0 and Kurtosis = 0.
```

Count For Categorial Data

Transform a metric variable into a quantitative one

cut expects a series of edge values used to cut the measurements or an integer number of groups used to cut the variables into equal-width bins

qcut expects a series of percentiles used to cut the variable

pd.cut(data, bins, ...):

Purpose: To bin data based on **fixed-width intervals**.
How it works: You provide the numerical data (a Series or array) and specify the bins.
The bins can be an integer (to create that many equally spaced bins between the minimum and maximum values) or a sequence of scalars defining the explicit bin edges.
Use Case: Use pd.cut() when you want to divide your data into bins of a specific size or based on predefined thresholds. The number of data points in each bin can vary significantly.
Example: Cutting ages into fixed 10-year intervals (0-10, 10-20, 20-30, etc.).

pd.qcut(data, q, ...):

Purpose: To bin data based on **quantiles**.
How it works: You provide the numerical data and specify q, the number of quantiles (e.g., 4 for quartiles, 10 for deciles). pd.qcut() determines the bin edges such that there is approximately the same number of observations in each bin.
Use Case: Use pd.qcut() when you want to distribute your data into bins with roughly an equal number of data points in each. The width of the intervals will vary.
Example: Cutting exam scores into quartiles so that roughly 25% of students fall into each quartile bin (e.g., top 25%, 25-50%, 50-75%, bottom 25%).

Modifying Data Distributions

- Obtain new feature creation from the combination of different but related variables

- Spot hidden groups or strange values lurking in your data

- Try some useful modifications of your data distributions by binning (or

  other discretizations such as binary variables)

- Let’s work on notebook week3-EDA.iynb

Summary

- dtypes inspection (dtypes_out = df.dtypes)
- Head & dtypes display
- Frequency distribution (e.g., Outcome)
- Descriptive stats for numerical columns + missing counts
- Bivariate analysis (group means, simple correlation)
- Boxplot (Glucose by Outcome) — matplotlib only
- Multivariate correlation heatmap
- Individual histograms (Glucose, BMI, Age)
- Skewness & kurtosis vs Normal (hist + PDF overlay)
- Binning: 4 fixed-width (Age) and 4 quantiles (BMI)Auto-generated
- “Summarize findings”

Python String Processing

- String is a container of characters; you can get a subset\ slice a string like any other Python container
- Slice from left or right side
- Slice by increments to get every other character form the string

  Check documentation for the complete list of String functions

  Pattern matching

- Let’s work on notebook during lab: week3-strings.iynb

Python Dates And Times

- Datetime library in Python
- String can be converted into a date Date formatting

  functions

- Date calculations

- Let’s work on notebook during lab: week3-datetime.iynb