Lecture 8:

- Summary stats (cont)
- Intro to data visualization

Announcements:

- HW-2 due
- HW-3 out, due next Thurs
- Quiz 4 on thurs (data preprocessing)
Summary Statistics

• Help discover what is in a given dataset
• Give (initial) insights into a dataset

1. Number of instances (how many rows in the table)

2. Number of distinct values in a column (attribute)

Q: Why might this be useful? For which measurement types?
  • often form nominal or ordinal, but potentially any discrete value
  • e.g., how many different origins are in the auto-mpg dataset?

3. Min and max attribute values

Q: Do these make sense for both all measurement types?
  • Ordinal, but not nominal  ... can only count number of each nominal value
  • Much easier if ordinal values are numeric!

Q: What should be done with missing values?
  • really, undefined / unknown
  • in practice just ignore them
4. “Middle” values of a distribution ... aka “central tendency”

Middle value: \((\text{max} + \text{min}) / 2.0\) ... aka midrange

(Arithmetic) Mean: \(\bar{x} = (x_1 + x_2 + \cdots + x_n)/n\) ... aka average

\[\text{Q: Problems with the mean?} \quad \text{... sensitive to extremes (e.g., outliers)}\]

\[\text{Q: For nominal and ordinal values?} \quad \text{... only interval, ratio (same widths)}\]

(Geometric) Mean: \(\bar{x} = \sqrt[n]{x_1 x_2 \cdots x_n}\)

- typically used for numbers that are meant to be multiplied (versus added)
- e.g., mean of \(0.5x\) (half as fast) and \(2x\) (twice as fast) is 1 (no speedup)

Median: the “middle” value in a set of sorted values

- if even number of values, halfway between the two middles
- or use “low” median variant (pick smaller of middle two)
- better measure than mean if data is “skewed”
- can be expensive for large data sets (sorting!)

Mode: value(s) that occurs most frequently

- typically assumes data is unimodal (one mode), e.g., normally distributed
- if not, can be useful to find each “mode” in multimodal data
5. Data Dispersion (Spread)

Range: \( \text{max} - \text{min} \)

Quantiles: (roughly) equal size partitions of data (if sorted smallest to largest)

- “2-quantiles” is the data point that divides into two halves
- “Quartiles” is three data points that divide into four groups
  - interquartile range is distance between 1st and 3rd quartiles
- “Percentiles” are 100-quantiles (100 groups)
- (Quartiles) used as part of boxplots (more later)

Variance: a measure of how spread out the data is

\[
\sum_{i=1}^{n} (x_i - \bar{x})^2 \frac{1}{n}
\]

- small variance implies data close to mean, large implies far from mean

Standard Deviation: square root of variance

- square root since variance is “average” of the \( n \) squares

For a normal (i.e., Gaussian) data distribution

- about 68% of values are within 1 standard deviation of mean
- about 95% of values are within 2 standard deviations
- about 99.7% of values are within 3 standard deviations
1. **Noisy vs Invalid Values**

Noisy: implies the value is correct, just recorded incorrectly
- e.g., decimal place error (5.72 instead of 57.2)
- or wrong categorical value used

Invalid: implies a noisy value that is not a valid value (for domain)
- e.g.: 57.2X, misspelled nominal value, value out of range (6 on 5-point scale)

Ways to deal with noisy and invalid values:
- look for duplicates (when there shouldn’t be)
- look for “outliers” (e.g., far from mean or large variance) ... entire subfield!
- sort and print range of (unique) values

The term “noisy” may also imply random error or random variance
- various techniques to “smooth out” values
- e.g., using means of bins or regression
2. Missing Values

Q: How to deal with these?

- Discard feature (column): kind of extreme, but maybe not useful feature
- Discard instances: throw out any row with a missing value
- Replace with a new value:
  - by hand
  - use a constant
  - use a central tendency measure (mean, median, most frequent, ...)
- most “probable” value (e.g., regression, using a classifier)
- replace either across data set, or based on similar instances

Q: What are possible issues with different approaches for missing values?

- biggest problem is introducing “bias” into data
- e.g., filled in value may be incorrect and end up skewing dataset

Note: Not all missing values may be bad (e.g., may just mean not applicable)
HW-3 pre-processing and exploration support functions

1. def distinct_values(table, column)
   • returns set of unique values in the given column of the table
   • can use Python sets to deal with duplicates
     
     ```python
     s = set()                        # create an empty set
     s.add(v)                        # add value v to the set (no change if v ∈ s)
     v in s                          # to check if value v is in the set
     ```

2. def remove_missing(table, columns)
   • removes all rows in the table with a missing value in one of the columns
   • where a missing value is represented as an empty string ("")

3. def duplicate_instances(table)
   • returns a data table with the duplicate rows in the given table
   • each duplicated row occurs only once in the result

4. def remove_duplicates(table)
   • removes the duplicate instances from the given table

5. def partition(table, columns)
   • returns a list of data tables whose rows form a partition of the original table
   • each partition (data table) has the same values for the given columns
   • this is similar to doing a "group by" – i.e., group rows by given columns