COMP9321 Data Services Engineering

Term3, 2019

Week 3: Data Cleansing and Manipulation

Agenda

- Understanding your data
- Data Cleansing
- Data Manipulation

Understanding the Data (ask the right Questions)

- What is this dataset?
- What should I expect within this dataset?
- Basic concepts (e.g., domain knowledge)
- What are the questions that I need to answer?
- Does the dataset have some sort of a schema? (utilize domain knowledge)

Understanding the Data using Python

- You can use the describe() function to get a summary about the data excluding the NaN values. This function returns the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.
- Use pandas .shape attribute to view the number of samples and features we're dealing with
- it's also a good idea to take a closer look at the data itself. With the help of the head() and tail() functions of the Pandas library, you can easily check out the first and last 5 lines of your DataFrame, respectively.
- Use pandas .sample attribute to view a random number of samples from the dataset

Data Cleansing

- Datasets are messy, messy data can give wrong insights (Martin Goodson's story*)
- Cleansing/Cleaning data "find and remove or correct data that detracts from the quality, and thus the usability, of data. The goal of data cleansing is to achieve consistent, complete, accurate, and uniform data"**

*https://www.martingoodson.com/ten-ways-your-data-project-is-going-to-fail/

** https://www.informatica.com/au/services-and-training/glossary-of-terms/data-cleansing-definition.html#fbid=eohTNIdjKaj

DB-hard Queries

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$406Bn
Microsoft	Redmond, WA	$392Bn
Intl. Business Machines	Armonk, NY	$194Bn

```
SELECT Market_Cap
From Companies
Where Company_Name = "Apple"
```

Number of Rows: 0

Problem:
Missing Data

DB-hard Queries

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$406Bn
Microsoft	Redmond, WA	$392Bn
Intl. Business Machines	Armonk, NY	$194Bn

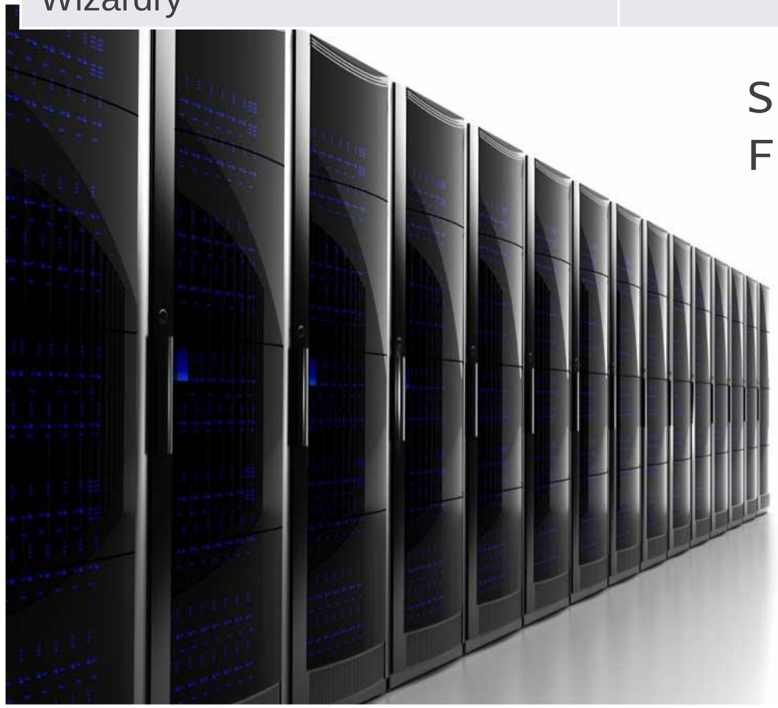
```
SELECT Market_Cap
From Companies
Where Company_Name = “IBM”
```

```
Number of Rows: 0
```

Problem:
Entity Resolution

DB-hard Queries

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$406
Microsoft	Redmond, WA	$392
Intl. Business Machines	Armonk, NY	$194
Hogwarts School of Witchcraft and Wizardry	Scotland, UK	$460



```
SELECT MAX(Market_Cap)
From Companies
```

Number of Rows: 1

Problem:
Unit Mismatch

Who’s Calling Who’S Data Dirty?

Dirty Data

The Statistics View:

- There is a process that produces data
- We want to model ideal samples of that process, but in practice we have non-ideal samples:
  - **Distortion** – some samples are corrupted by a process
  - **Selection Bias** - likelihood of a sample depends on its value
  - **Left and right censorship** - users come and go from our scrutiny
  - **Dependence** – samples are supposed to be independent, but are not (e.g. social networks)
- You can add new models for each type of imperfection, but you can't model everything.
- What's the best trade-off between accuracy and simplicity?

Dirty Data

The Database View:

- I got my hands on this data set
- Some of the values are missing, corrupted, wrong, duplicated
- Results are absolute (relational model)
- You get a better answer by improving the quality of the values in your dataset

Dirty Data

The Domain Expert's View:

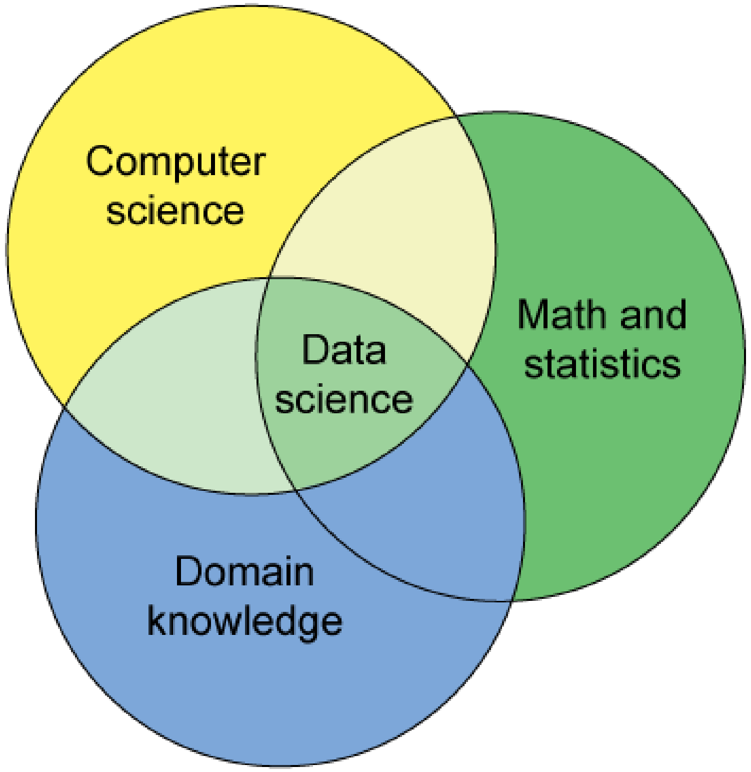
- This Data Doesn't look right
- This Answer Doesn't look right
- What happened?

Domain experts have an implicit model of the data that they can test against…

Dirty Data

The Data Scientist's View:

- Some Combination of all of the above



Example: Data Quality Problems

Representation

Contradictions

Ref. integrity

CUST

CNr	Name	Birthday	Age	Sex	Phone	ZIP
1234	Costa, Rui	18.2.80	37	m	999999999	1000
1234	Ana Costa	32.2.70	37	m	965432123	55555
1235	Rui Costa	18.2.80	27	m	963124568	1000

Uniqueness

Missing values

Duplicates

ADDRESS

ZIP	Place
1000	Lisboa
1000	Lsiboa
1024	Portugal

Typos

Incorrect values

Data Quality Problems

- (Source) Data is dirty on its own.
- Transformations corrupt the data (complexity of software pipelines).
- Data sets are clean but integration (i.e., combining them) screws them up.
- “Rare” errors can become frequent after transformation or integration.
- Data sets are clean but suffer “bit rot”
- Old data loses its value/accuracy over time
- Any combination of the above

Why Data Quality Problems Matter

Incorrect prices in inventory retail databases

- Costs for consumers 2.5 billion $
- 80% of barcode-scan-errors to the disadvantage of consumer

IRS 1992: almost 100,000 tax refunds not deliverable

- 50% to 80% of computerized criminal records in the U.S. were found to be inaccurate, incomplete, or ambiguous. [Strong et al. 1997a]

US-Postal Service: of 100,000 mass mailings up to 7,000 undeliverable due to incorrect addresses [Pierce 2004]

… …

How Data Quality Problems Happen

Incomplete data comes from:

 non available data value when collected

 different criteria between the time when the data was collected and when it is analyzed

 human/hardware/software problems

Noisy data comes from:

 data collection: faulty instruments

 data entry: human or computer errors

 data transmission

Inconsistent (and duplicate) data comes from:

 Different data sources, so non-uniform naming conventions/data codes

 Functional dependency and/or referential integrity violation

Application Scenarios

Integrate data from different sources

- E.g., populating data from different operational data stores or a mediator-based architecture

Eliminate errors and duplicates within a single source

- E.g., duplicates in a file of customers

Migrate data from a source schema into a different fixed target schema

- E.g., discontinued application packages

Convert poorly structured data into structured data

- E.g., processing data collected from the Web

Why Data Cleaning is Important

Activity of converting source data into target data without errors, duplicates, and inconsistencies, i.e., Cleaning and Transforming to get…

- **High-quality data!**

No quality data, no quality decisions!

- Quality decisions must be based on good quality data (e.g., duplicate or missing data may cause incorrect or even misleading statistics)

Data Quality Problems

Schema level data quality problems

- prevented with better schema design, schema translation and integration.

Instance level data quality problems

- errors and inconsistencies of data that are not prevented at schema level

Data Quality Problems

Schema level data quality problems

- Avoided by an RDBMS
  - Missing data – *product price not filled in*
  - Wrong data type – *“abc” in product price*
  - Wrong data value – *0.5 in product tax (iva)*
  - Dangling data – *category identifier of product does not exist*
  - Exact duplicate data – *different persons with same ssn*
  - Generic domain constraints – *incorrect invoice price*
- Not avoided by an RDBMS
  - Wrong categorical data – *countries and corresponding states*
  - Outdated temporal data – *just-in-time requirement*
  - Inconsistent spatial data – *coordinates and shapes*
  - Name conflicts – *person vs person or person vs client*
  - Structural Conflicts - *addresses*

Data Quality Problems

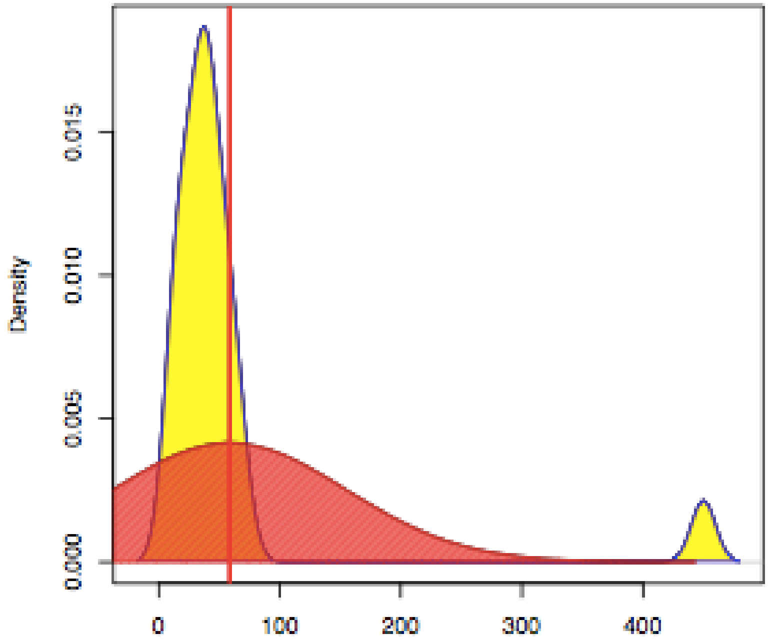
Instance level data quality problems

- Single record
  - Missing data in a *not null* field – *ssn:-9999999*
  - Erroneous data – *price:5 but real price:50*
  - Misspellings: *José Maria Silva vs José Maria Sliva*
  - Embedded values: *Prof. José Maria Silva*
  - Misfielded values: *city: Portugal*
  - Ambiguous data: *J. Maria Silva; Miami Florida,Ohio*
- Multiple records
  - Duplicate records: *Name:Jose Maria Silva, Birth:01/01/1950 and Name:José Maria Sliva, Birth:01/01/1950*
  - Contradicting records: *Name:José Maria Silva, Birth:01/01/1950 and Name:José Maria Silva, Birth:01/01/1956*
  - Non-standardized data: *José Maria Silva vs Silva, José Maria*

Numeric Outliers

12	13	14	21	22	26	33	35	36	37	39	42	45	47	54	57	61	68	450

ages of employees (US)



- median 37
- mean 58.52632
- variance 9252.041

Adapted from Joe Hellerstein's 2012 CS 194 Guest Lecture

Integration error

Data 1

…	Date(mm/dd/yyyy)	…
…	08/02/2019	…
…	09/02/2019	…

Data 2

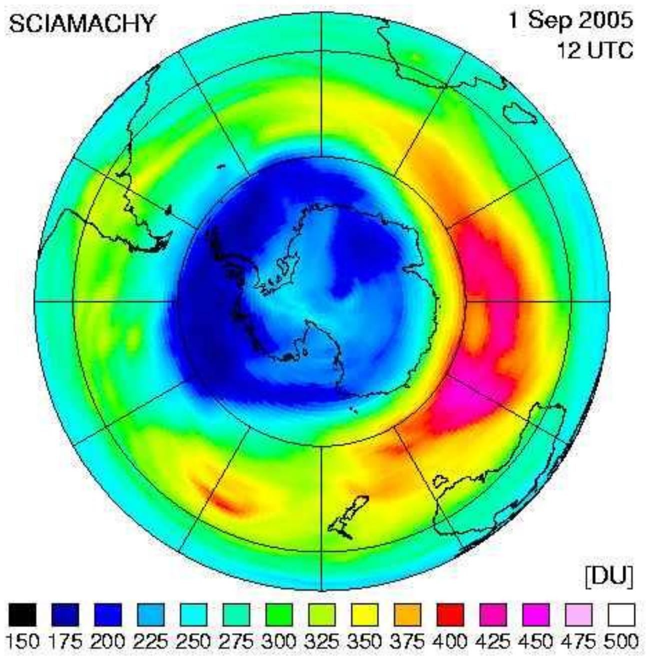
…	Date(dd/mm/yyyy)	…
…	08/08/2019	…
…	09/08/2019	…

…	Date(mm/dd/yyyy)	…
…	08/02/2019	…
…	09/02/2019	…
…	08/08/2019	…
…	09/08/2019	…

Data Cleaning Makes Everything Okay?

The appearance of a hole in the earth's ozone layer over Antarctica, first detected in 1976, was so unexpected that scientists didn't pay attention to what their instruments were telling them;

they thought their instruments were malfunctioning.



In fact, the data were rejected as unreasonable by data quality control algorithms

Conventional Definition of Data Quality

Accuracy

- The data was recorded correctly.

Completeness

- All relevant data was recorded.

Uniqueness

- Entities are recorded once.

Timeliness

- The data is kept up to date.
  - Special problems in federated data: time consistency.

Consistency

- The data agrees with itself.

Accuracy

Closeness between a value v and a value v'

considered as the correct representation of the realworld phenomenon that v aims to represent.

- Ex: for a person name "John", v'=John is correct, v=Jhn is incorrect

Syntatic accuracy

closeness of a value v to the elements of the corresponding definition domain D

- Ex: if v=Jack, even if v'=John , v is considered syntactically correct because it is an admissible value in the domain of people names.
- Measured by means of comparison functions (e.g., edit distance) that returns a score

Accuracy

Semantic accuracy

closeness of the value v to the true value v’

- Measured with a <yes, no> or <correct, not correct> domain
- Coincides with correctness
- The corresponding true value has to be known

Ganularity of accuracy definition

Accuracy may refer to:

- a single value of a relation attribute
- an attribute or column
- a relation
- the whole database

Completeness

"The extent to which data are of sufficient breadth, depth, and scope for the task in hand."

Three types:

- **Schema completeness**: degree to which concepts and their properties are not missing from the schema
- **Column completeness**: evaluates the missing values for a specific property or column in a table.
- **Population completeness**: evaluates missing values with respect to a reference population

Completeness of relational data

The **completeness of a table** characterizes the extent to which the table represents the real world.

- **The presence/absence and meaning of null values**

Example: Person(name, surname, birthdate, email), if email is null may indicate the person has no mail (no incompleteness), email exists but is not known (incompletenss), is is not known whether Person has an email (incompleteness may not be the case)

Completeness of relational data

- **Validity of open world assumption (OWA) or closed world assumption (CWA)**
  - OWA: cannot state neither the truth or falsity of facts not represented in the tuples of a relation
  - CWA: only the values actually present in a relational table and no other values represent facts of the real world.

Example

```
Statement: "Mary" "is a citizen of" "France"
```

```
Question: Is Paul a citizen of France?
```

```
"Closed world" (for example SQL) answer: No.
"Open world" answer: Unknown.
```

Time-related Dimensions

Currency:

concerns how promptly data are updated

- Example:
  - if the residential address of a person is updated (it corresponds to the address where the person lives) then the currency is high

Volatility:

characterizes the frequency with which data vary in time

- Example:
  - Birth dates (volatility zero) vs stock quotes (high degree of volatility)

Time-related Dimensions

Timeliness

- expresses how current data are for the task in hand

- Example:
  - The timetable for university courses can be current by containing the most recent data, but it cannot be timely if it is available only after the start of the classes.

Consistency

Captures the violation of semantic rules defined over a set of data items, where data items can be tuples of relational tables or records in a file

- Integrity constraints in relational data
  - Domain constraints, Key, inclusion and functional dependencies
- Data edits: semantic rules in statistics

Others

- Interpretability: concerns the documentation and metadata that are available to correctly interpret the meaning and properties of data sources
- Synchronization between different time series: concerns proper integration of data having different time stamps.
- Accessibility: measures the ability of the user to access the data from his/her own culture, physical status/functions, and technologies available.

Problems …

Unmeasurable

- Accuracy and completeness are extremely difficult, perhaps impossible to measure.

Context independent

- No accounting for what is important. E.g., if you are computing aggregates, you can tolerate a lot of inaccuracy.

Vague

- The conventional definitions provide no guidance towards practical improvements of the data.

Data Cleansing (Practical Approach)

- Dealing with Missing Data
- Removing Unnecessary Data (rows, or columns)
- Normalizing/Formatting data

Dealing with Missing Data

- One of the most common problems is missing data. This could be because it was never filled out properly, the data wasn’t available, or there was a computing error. Whatever the reason, if we leave the blank values in there, it will cause errors in analysis later on. There are few things to cinsider when dealing with missing values:

  - Are we dealing with Standard missing values
  - Are we dealing with Standard missing values
  - Are We dealing with unexpected missing values.

*http://www.developintelligence.com/blog/2017/08/data-cleaning-pandas-python/

Dealing with Standard Missing Data

ST_NUM	ST_NAME	NUM_BEDROOMS	OWN_OCCUPIED
104	PUTNAM	3	Y
197	LEXINGTON	3	N
	LEXINGTON	n/a	N
201	BERKELEY	1	12
203	BERKELEY	3	Y
207	BERKELEY	NA	Y
NA	WASHINGTON	2	
213	TREMONT	--	Y
215	TREMONT	na	Y

https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Dealing with Non Standard Missing Data

ST_NUM	ST_NAME	NUM_BEDROOMS	OWN_OCCUPIED
104	PUTNAM	3	Y
197	LEXINGTON	3	N
	LEXINGTON	n/a	N
201	BERKELEY	1	12
203	BERKELEY	3	Y
207	BERKELEY	NA	Y
NA	WASHINGTON	2	
213	TREMONT	--	Y
215	TREMONT	na	Y

https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Dealing with Non Standard Missing Data with Pandas

- You can make a list of possible presentations of missing values

```
missing_values = ["n/a", "na", "--“, “-”, “None”]
df = pd.read_csv(“myfile.csv", na_values = missing_values)
```

*http://www.developintelligence.com/blog/2017/08/data-cleaning-pandas-python/

Dealing with Unexpected Missing Data

ST_NUM	ST_NAME	NUM_BEDROOMS	OWN_OCCUPIED
104	PUTNAM	3	Y
197	LEXINGTON	3	N
	LEXINGTON	n/a	N
201	BERKELEY	1	12
203	BERKELEY	3	Y
207	BERKELEY	NA	Y
NA	WASHINGTON	2	
213	TREMONT	--	Y
215	TREMONT	na	Y

https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Dealing with Unexpected Missing Data with Pandas

- Looping through the column can help

```
# Detecting numbers
cnt=0
for row in df['OWN_OCCUPIED']:
    try:
        int(row)
        df.loc[cnt, 'OWN_OCCUPIED']=np.nan
    except ValueError:
        pass
    cnt+=1
```

https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Summarizing Missing Data

- After Cleaning the data we probably need to summarize the data to see what we are dealing with

```
print df.isnull().sum()
Out:
ST_NUM 2
ST_NAME 0
OWN_OCCUPIED 2
NUM_BEDROOMS 4

# Total number of missing values
print df.isnull().sum().sum()Out:
8
```


https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Replacing Missing Data

- You might need to replace the missing data

```
# Replace missing values with a number
df['ST_NUM'].fillna(125, inplace=True)
```

```
# Replace using median
median = df['NUM_BEDROOMS'].median()
df['NUM_BEDROOMS'].fillna(median, inplace=True)
```


https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b

Removing Unnecessary Data

- Some times you don’t need all the data in the tables so it might help you achieve better performance if you remove the irrelevant data.
- Some columns or rows might be useless for you in the analysis due to having many missing values and replacing them with default values would produce wrong insights.
- Python has a very good function Drop() to help you with this

Removing Unnecessary Data Example

- Dropping Columns with all NaN values

Example: data.dropna(axis=1, how=’all’)

- Dropping Raws with all NaN values

Example: data.dropna(axis=0, how=’all’) # you don’t need to have axis=0

- Dropping Multiple Columns

to_drop = [Column1’, Column6’,Column12’, Column13’, ' Column14 ',' Column16’, Column17 ‘, 'Column18’]

data.drop(to_drop, inplace=True, axis=1)

*http://www.developintelligence.com/blog/2017/08/data-cleaning-pandas-python/

** https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.fillna.html

Normalizing/ Formatting data

- Data read from source may not have the correct format (e.g., reading integer as a string)
- Some strings in the data have spacing which might not play well with your analysis at some point.
- The date/time format may not appropriate for your analysis
- Some times the data is generated by a computer program, so it probably has some computer-generated column names, too. Those can be hard to read and understand while working.

Normalizing/ Formatting data Examples

- Example1 (change data type on read):

df = pd.read_csv(‘mydata.csv’, dtype={‘Integer_Column’: int})

- Example2 (change data type in dataframe)

df['column'] = df['column'].to_numeric()

df['column'] = df['column'].astype(str)

- Example3 (Spacing within the values):

data[‘Column_with_spacing’].str.strip()

Normalizing/ Formatting data Examples

- Example4 (unnecessary time item in the date field):

df[‘MonthYear’] = pd.to_datetime(df[‘MonthYear’])

df[‘MonthYear’] = df[‘MonthYear’].apply(lambda x: x.date())

- Example5 (rename columns)

data = data.rename(columns = {‘Bad_Name1’:Better_Name1’, ‘Bad_Name2’:’Better_name2’})

Useful Read

- Python for Data Analysis, Wes McKinney
- https://www.altexsoft.com/blog/datascience/preparing-your-dataset-for-machine-learning-8-basic-techniques-that-make-your-data-better/
- https://pandas.pydata.org/pandas-docs/stable/tutorials.html
- https://www.analyticsvidhya.com/blog/2016/01/12-pandas-techniques-python-data-manipulation/
- https://www.dataquest.io/blog/machine-learning-preparing-data/