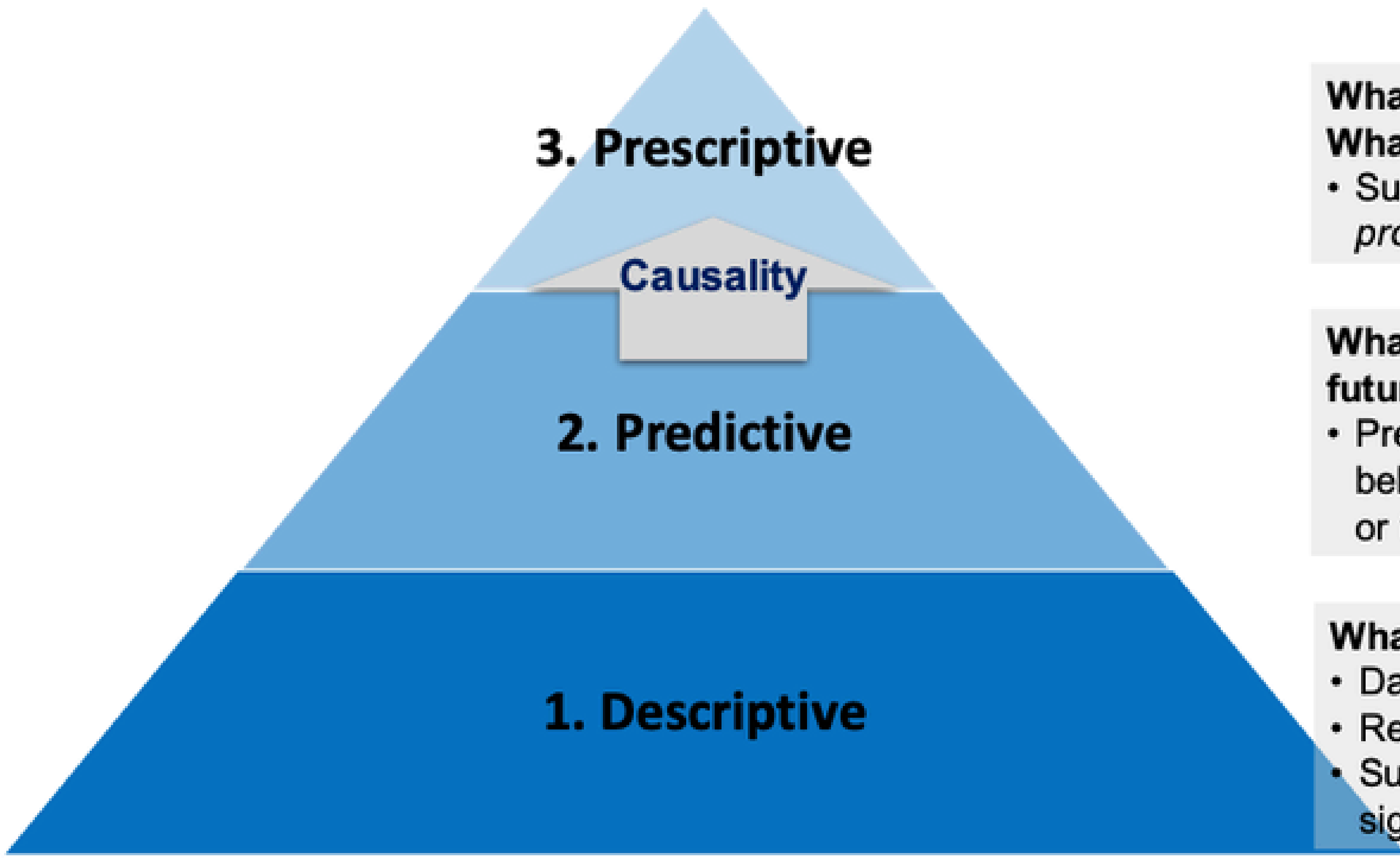
Exploratory Data Analysis (EDA)

J Andres Gannon

January 25, 2025

Types of data science



What should we do?
What is the Best Decision?
- Support *decision making* and *proactive* actions

What will happen in the future?
- Predict forward-looking behavior, events, probabilities, or trends

What happened in the past?
- Data visualization
- Reports and profiling
- Summary statistics & significance testing

Purpose of EDA

1. Communicate - present data and explain and inform with evidence
2. Analyze - explore data to assess a situation and determine how to proceed

Descriptive statistics do this by identifying:

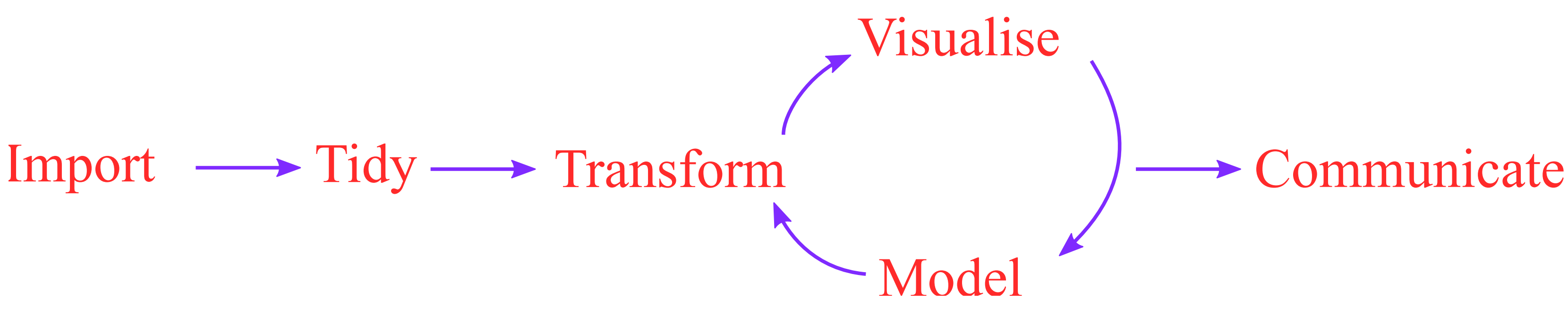
- Kinds of values
- Outliers (possibly incorrect)
- Distribution (possibly skewed)

Two basic EDA principles

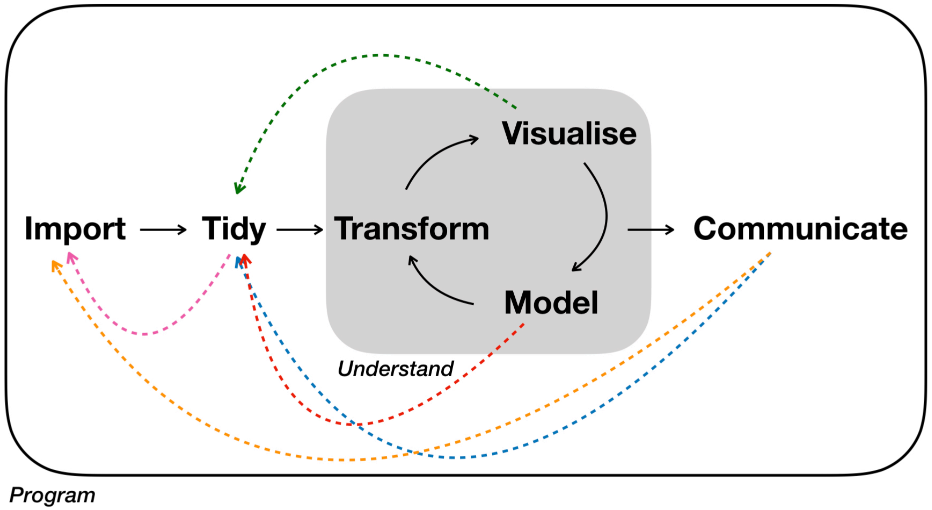
1. Making a simpler description possible is good
2. Looking one level below an existing description is good

So we shall always be glad (a) to simplify description and (b) to describe one layer deeper.

Where exploratory data analysis fits in



Where exploratory data analysis fits in (reality)



Understand data via single variable, pair of variables, or dimensionality reduction

Organizes the data, spots problems, and identifies modeling strategies

How to describe a dataset

Key descriptive statistics

```python
import pandas as pd
import numpy as np
df = pd.read_csv("https://github.com/nlihin/data-analytics/raw/main/dataset
df.describe()
```

	new_cases	new_deaths	new_tests
count	248.000000	248.000000	135.000000
mean	1094.818548	143.133065	31699.674074
std	1554.508002	227.105538	11622.209757
min	-148.000000	-31.000000	7841.000000
25%	123.000000	3.000000	25259.000000
50%	342.000000	17.000000	29545.000000

	new_cases	new_deaths	new_tests
75%	1371.750000	175.250000	37711.000000
max	6557.000000	971.000000	95273.000000

Key descriptive statistics

Low variance means values close to the mean

```
df.var(numeric_only = True)
```

```
new_cases      2.416495e+06
new_deaths     5.157693e+04
new_tests      1.350758e+08
dtype: float64
```

Skewness is symmetry of the data. Positive skew indicates large outliers. Negative skew indicates small outliers

```
df.skew(numeric_only = True)
```

```
new_cases      1.728277
new_deaths     1.703742
new_tests      1.619825
dtype: float64
```

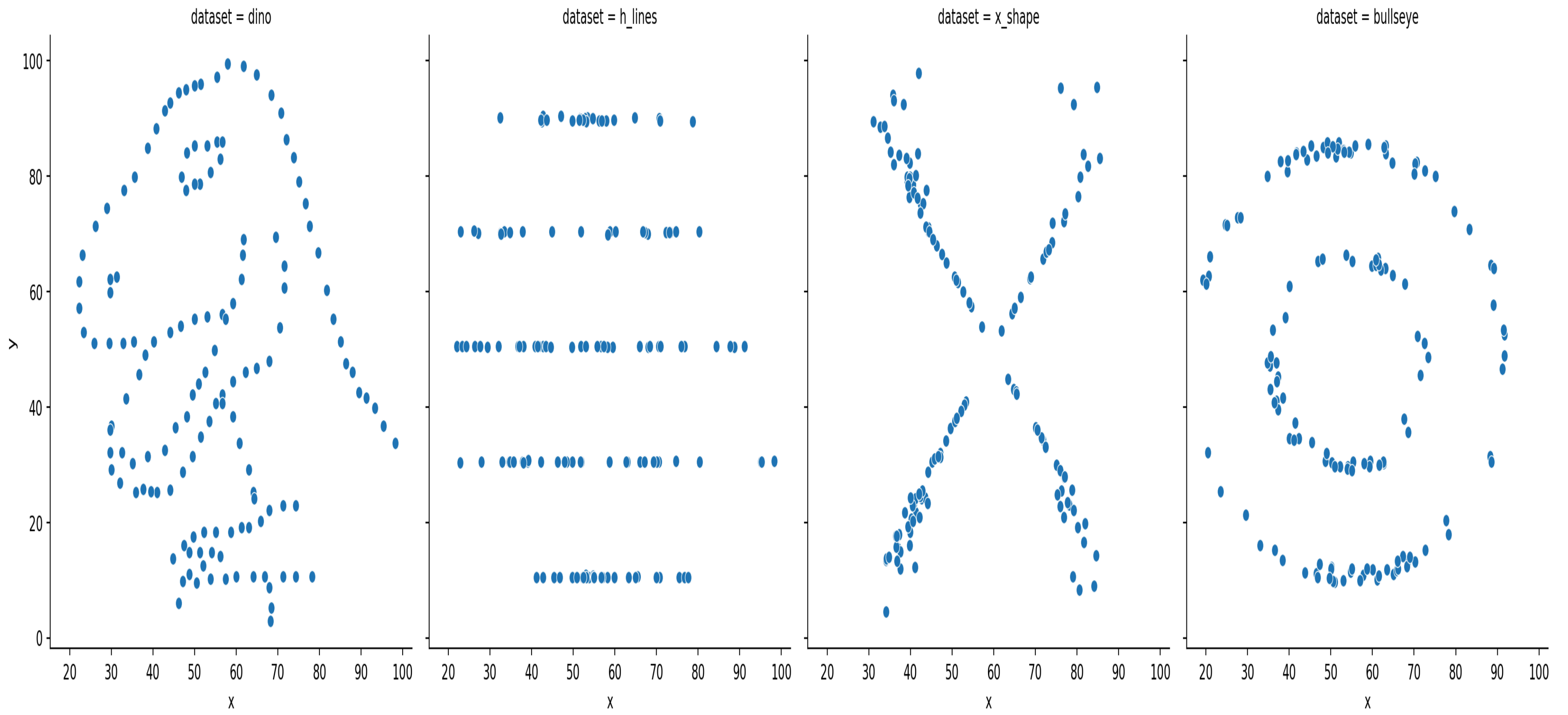
Descriptive statistics can mislead

```python
datasaurus_data = pd.read_csv('./data/datasaurus.csv')
datasaurus_data.groupby('dataset').agg({'x': ['count','mean', 'std'],'y': [
```

	x			y	
	count	mean	std	count	mean
dataset					
bullseye	142	54.268730	16.769239	142	47.830823
dino	142	54.263273	16.765142	142	47.832253
h_lines	142	54.261442	16.765898	142	47.830252
x_shape	142	54.260150	16.769958	142	47.839717

Descriptive statistics can mislead

```python
import seaborn as sns
sns.relplot(data=datasaurus_data, x='x', y='y', col='dataset', col_wrap=4)
```



Trust nothing and no one

BBC Sign in Home News Sport Reel Worklife

NEWS

Home | War in Ukraine | Coronavirus | Climate | Video | World | US & Canada | UK | Business | Tech

Tech

Excel: Why using Microsoft's tool caused Covid-19 results to be lost

SCIENCE / TECH / MICROSOFT

Scientists rename human genes to stop Microsoft Excel from misreading them as dates

SCHENECTADY COUNTY

NYCLU walks back report on pot arrests

Civil liberties group apologizes to police for exaggerated numbers

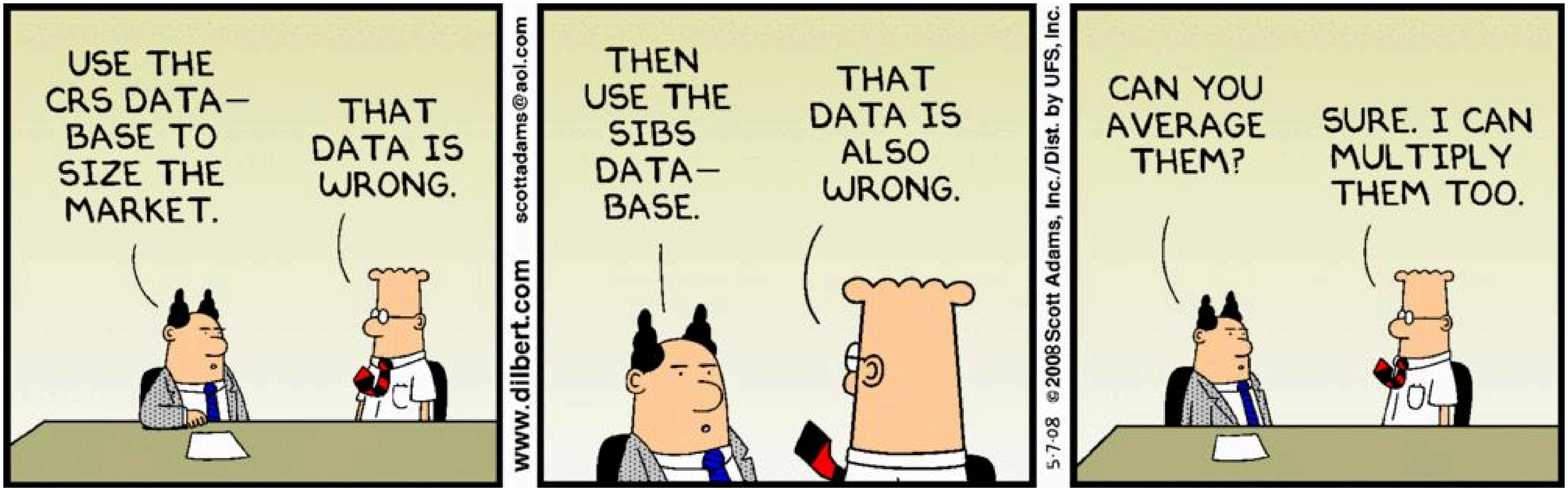
By Pete DeMola | July 5, 2019

Why missingness happens

Column values missing

- Subject-created: opt out, unknown value
- Record-created: entry, software, or coding errors

No universal solution, but many wrong ones



Computational strategies

Guiding questions

- How much missingness is present
- Is missing value in response variable or predictor variable
- Is missing value quantitative or categorical
- Specifying allowed NA values when reading in data
- Look for nonsense values (outlier check)
- Missingness when joining/merging

Mean Imputation

```python
df = pd.read_csv("./data/titanic.csv")
df = df[["PassengerId", "Survived", "Name", "Sex", "Age", "Class"]]

# Mean imputate for numeric
df_imputed = df
df_imputed['Age'] = df_imputed['Age'].fillna(df_imputed['Age'].mean())
df_imputed.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  1309 non-null   int64  
 1   Survived     891 non-null    float64
 2   Name         1309 non-null   object 
 3   Sex          1309 non-null   object 
 4   Age          1309 non-null   float64
 5   Class        1304 non-null   float64
dtypes: float64(3), int64(1), object(2)
memory usage: 61.5+ KB
```

Multiple Imputation: Random Sample

Replace each missing value with a random value from that column

```
df = pd.read_csv("./data/titanic.csv")
df = df[["PassengerId", "Survived", "Name", "Sex", "Age", "Class"]]

df['Age'].dropna().sample()
```

```
777    5.0
Name: Age, dtype: float64
```

Multiple Imputation: k-nearest neighbor

K-nearest neighbor: impute from rows that most closely match based on non-missing variables

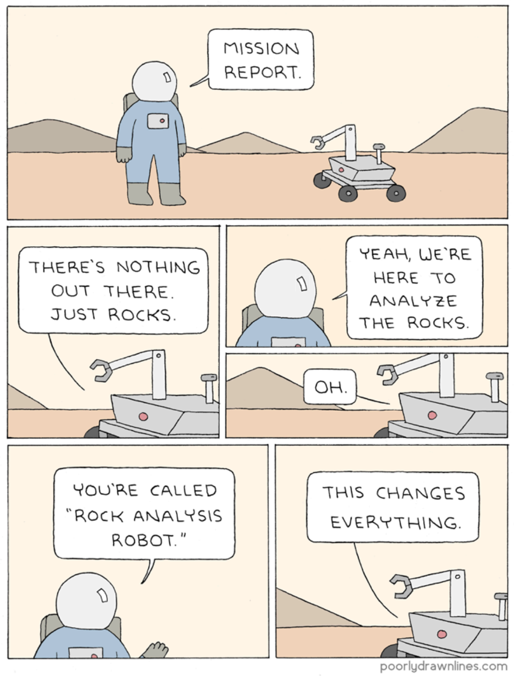
Requires numeric columns and normalization

```
from sklearn.impute import KNNImputer

df = pd.read_csv("./data/titanic.csv")
df = df[["PassengerId", "Survived", "Age", "Class"]]

imp = KNNImputer(n_neighbors = 2, weights = "uniform")
df = pd.DataFrame(imp.fit_transform(df), columns = df.columns)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype
```

```
---  ------       --------------  -----
 0   PassengerId  1309 non-null   float64
 1   Survived     1309 non-null   float64
 2   Age          1309 non-null   float64
 3   Class        1309 non-null   float64
dtypes: float64(4)
memory usage: 41.0 KB
```

EDA can seem mundane…

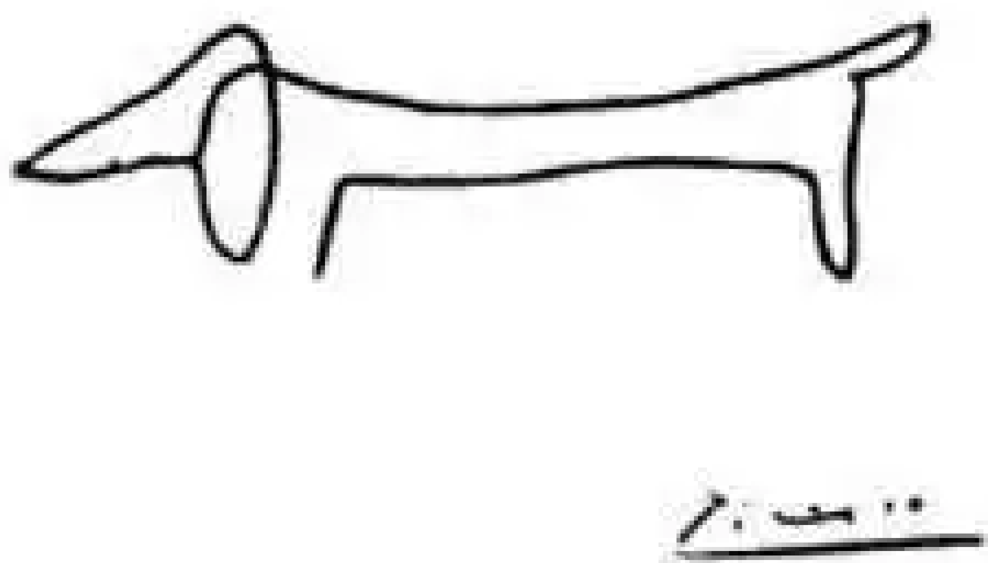


90% cleaning and documentation

9% existing off the shelf tools

1% cutting edge

…but it's the foundation of everything else



State of the art (pun intended)