dm25s1

Topic 05 : Exploratory Data Analysis2

Part 01 : EDA Pass3

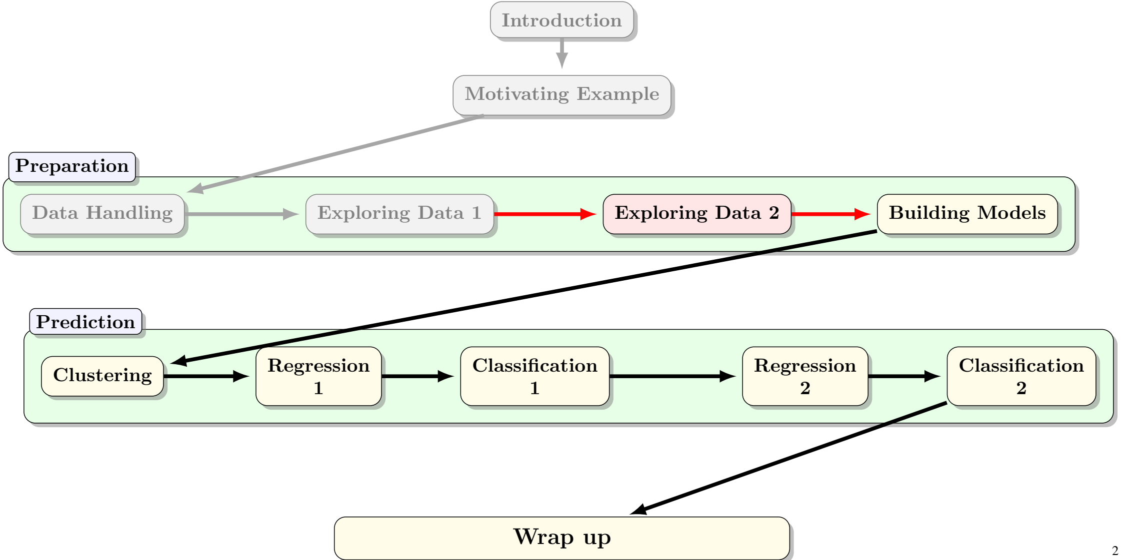
Dr Bernard Butler
Department of Computing and Mathematics, WIT.
(bernard.butler@setu.ie)

Autumn Semester, 2025

Outline

- What is EDA Pass 3?
- Datasets = Tips, Titanic and Algae Blooms
- Identifying and resolving issues (missing value, outliers)

Data Mining (Week 5)



EDA Pass3 — Summary

Acknowledgment

A big thanks to Dr Kieran Murphy, who provided some of the slides for today's lecture.

Categorical vs Numerical Variables

Recall statistical *Levels of Measurement*

Categorical vs Numerical Variables

Recall statistical *Levels of Measurement*

Type	Drawn from	Examples	Used to
Nominal	Finite Set, Unrelated	Manufacturers, Countries, Gender	Categorise with descriptive label
Ordinal	Finite Set, Ordered	Size (S,M,L), Army Ranks, Satisfaction	Categorise with descriptive label
Interval	Ordered, Differences matter	Exam Scores, Temperatures (Celsius)	Assign numeric score to
Ratio	Ordered, Differences and ratios matter	Distance (m), Cost ($), Temperatures (Kelvin)	Assign numeric score to

Categorical vs Numerical Variables

Recall statistical *Levels of Measurement*

Type	Drawn from	Examples	Used to
Nominal	Finite Set, Unrelated	Manufacturers, Countries, Gender	Categorise with descriptive label
Ordinal	Finite Set, Ordered	Size (S,M,L), Army Ranks, Satisfaction	Categorise with descriptive label
Interval	Ordered, Differences matter	Exam Scores, Temperatures (Celsius)	Assign numeric score to
Ratio	Ordered, Differences and ratios matter	Distance (m), Cost ($), Temperatures (Kelvin)	Assign numeric score to

Generally, *Nominal* and *Ordinal* are considered categorical, *Interval* and *Ratio* are considered Numerical

Categorical vs Numerical Variables

Recall statistical *Levels of Measurement*

Type	Drawn from	Examples	Used to
Nominal	Finite Set, Unrelated	Manufacturers, Countries, Gender	Categorise with descriptive label
Ordinal	Finite Set, Ordered	Size (S,M,L), Army Ranks, Satisfaction	Categorise with descriptive label
Interval	Ordered, Differences matter	Exam Scores, Temperatures (Celsius)	Assign numeric score to
Ratio	Ordered, Differences and ratios matter	Distance (m), Cost ($), Temperatures (Kelvin)	Assign numeric score to

Generally, *Nominal* and *Ordinal* are considered categorical, *Interval* and *Ratio* are considered Numerical

But what about a variable which contains *Months of the year*?

First Pass—Load Dataset and Initial Clean

- Load dataset

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general "table" format
  - Can be local file or url (read over network)

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types
  - Convert strings to categories, possibly grouping, where possible

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general "table" format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types
  - Convert strings to categories, possibly grouping, where possible
  - Ensure numeric data is stored as number (watch out for "Unknown" etc.)

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general "table" format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types
  - Convert strings to categories, possibly grouping, where possible
  - Ensure numeric data is stored as number (watch out for "Unknown" etc.)
- Identify (and possibly address) missing values

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types
  - Convert strings to categories, possibly grouping, where possible
  - Ensure numeric data is stored as number (watch out for “Unknown” etc.)
- Identify (and possibly address) missing values
  - Missing values by row or column

First Pass—Load Dataset and Initial Clean

- Load dataset
  - Typically either csv or more general “table” format
  - Can be local file or url (read over network)
- Check variables names
  - Should be meaningful and distinct
  - Avoid clashes with reserved words (python or statistical)
- Verify variable types
  - Convert strings to categories, possibly grouping, where possible
  - Ensure numeric data is stored as number (watch out for “Unknown” etc.)
- Identify (and possibly address) missing values
  - Missing values by row or column
  - Leave blank, impute value, drop row/column?

Second Pass — Individual Features and Target

- Categorical vs numerical target

Second Pass — Individual Features and Target

- Categorical vs numerical target

- Categorical vs numerical features
- Identify (and possibly address) issues

Is it usable?

Second Pass — Individual Features and Target

- Categorical vs numerical target

- Categorical vs numerical features
- Identify (and possibly address) issues

Is it usable?

- Relationship to target.

Is it useful?

Dataset: `Titanic`, Target: `Survived`

```
df.Survived.describe()
```

```
count     891
unique      2
top         0
freq      549
Name: Survived, dtype: int64
```

Dataset: `Titanic`, Target: `Survived`

```
df.Survived.describe()
```

```
count    891
unique     2
top        0
freq     549
Name: Survived, dtype: int64
```

```
df.Survived.unique()
```

```
[0, 1]
Categories (2, int64): [0, 1]
```

Dataset: `Titanic`, Target: `Survived`

```
df.Survived.value_counts(normalize=True, dropna=False)
```

```
Survived
0    0.616162
1    0.383838
Name: proportion, dtype: float64
```

```
df.Survived.describe()
```

```
count     891
unique      2
top         0
freq      549
Name: Survived, dtype: int64
```

```
df.Survived.unique()
```

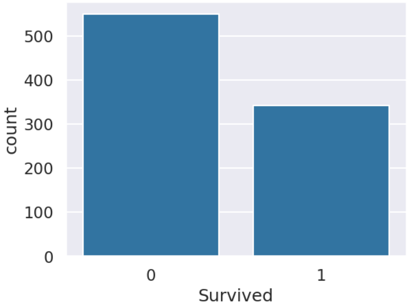
```
[0, 1]
Categories (2, int64): [0, 1]
```

Dataset: Titanic, Target: Survived

```
df.Survived.value_counts(normalize=True, dropna=False)
```

```
Survived
0    0.616162
1    0.383838
Name: proportion, dtype: float64
```

```
sns.countplot(x="Survived", data=df);
```




```
df.Survived.describe()
```

```
count     891
unique      2
top         0
freq      549
Name: Survived, dtype: int64
```

```
df.Survived.unique()
```

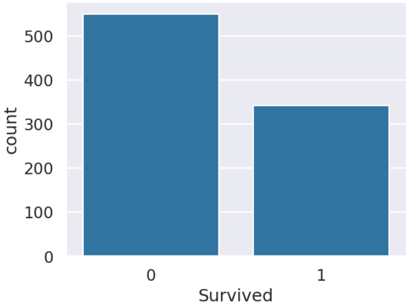
```
[0, 1]
Categories (2, int64): [0, 1]
```

Dataset: Titanic, Target: Survived

```
df.Survived.value_counts(normalize=True, dropna=False)
```

```
Survived
0    0.616162
1    0.383838
Name: proportion, dtype: float64
```

```
sns.countplot(x="Survived", data=df);
```




```
df.Survived.describe()
```

```
count     891
unique      2
top         0
freq      549
Name: Survived, dtype: int64
```

```
df.Survived.unique()
```

```
[0, 1]
Categories (2, int64): [0, 1]
```

- Simplest classification problem (two classes) with both classes nearly equal frequency.
- In a unbalanced classification problem where the minority class occurs about 20% or lower, models can focus on the majority class.

Dataset: Algae Blooms, Target: a1, . . . , a7

```
targets = [c for c in df.columns if c[0]=="a"]
targets
```

```
['a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'a7']
```

Dataset: `Algae Blooms`, Target: `a1`, . . . , `a7`

```
targets = [c for c in df.columns if c[0]=="a"]
targets
```

```
['a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'a7']
```

```
df[targets].describe()
```

	a1	a2	a3	a4	a5	a6	a7
count	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000
mean	16.996465	7.470707	4.334343	1.997475	5.115657	6.004545	2.487374
std	21.421713	11.065461	6.976788	4.439205	7.511846	11.711053	5.181536
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.525000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	6.950000	3.000000	1.550000	0.000000	2.000000	0.000000	1.000000
75%	24.800000	11.275000	4.975000	2.400000	7.500000	6.975000	2.400000
max	89.800000	72.600000	42.800000	44.600000	44.400000	77.600000	31.600000

Dataset: Algae Blooms, Target: a1, . . . , a7

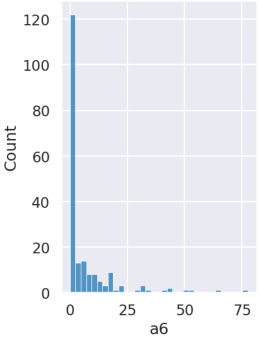
```
targets = [c for c in df.columns if c[0]=="a"]
targets
```

```
['a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'a7']
```

```
df[targets].describe()
```

	a1	a2	a3	a4	a5	a6	a7
count	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000
mean	16.996465	7.470707	4.334343	1.997475	5.115657	6.004545	2.487374
std	21.421713	11.065461	6.976788	4.439205	7.511846	11.711053	5.181536
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.525000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	6.950000	3.000000	1.550000	0.000000	2.000000	0.000000	1.000000
75%	24.800000	11.275000	4.975000	2.400000	7.500000	6.975000	2.400000
max	89.800000	72.600000	42.800000	44.600000	44.400000	77.600000	31.600000

```
plt.figure(figsize=(4,6))
sns.histplot(x="a6", data=df);
```



Dataset: `Algae Blooms`, Target: `a1`, . . . , `a7`

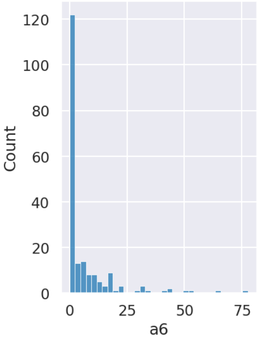
```
targets = [c for c in df.columns if c[0]=="a"]
targets
```

```
['a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'a7']
```

```
df[targets].describe()
```

	a1	a2	a3	a4	a5	a6	a7
count	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000	198.000000
mean	16.996465	7.470707	4.334343	1.997475	5.115657	6.004545	2.487374
std	21.421713	11.065461	6.976788	4.439205	7.511846	11.711053	5.181536
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.525000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	6.950000	3.000000	1.550000	0.000000	2.000000	0.000000	1.000000
75%	24.800000	11.275000	4.975000	2.400000	7.500000	6.975000	2.400000
max	89.800000	72.600000	42.800000	44.600000	44.400000	77.600000	31.600000

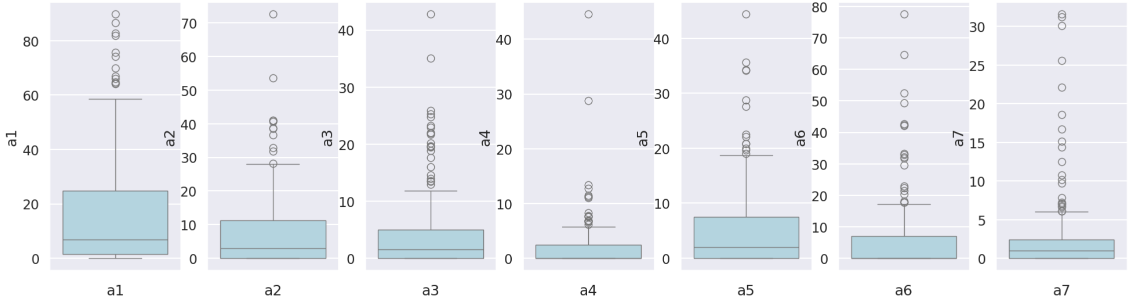
```
plt.figure(figsize=(4,6))
sns.histplot(x="a6", data=df);
```



All distributions are heavily skewed to the right, many with outliers (see next slide). All of the zero measurements are probably due to population levels too low to be measured.

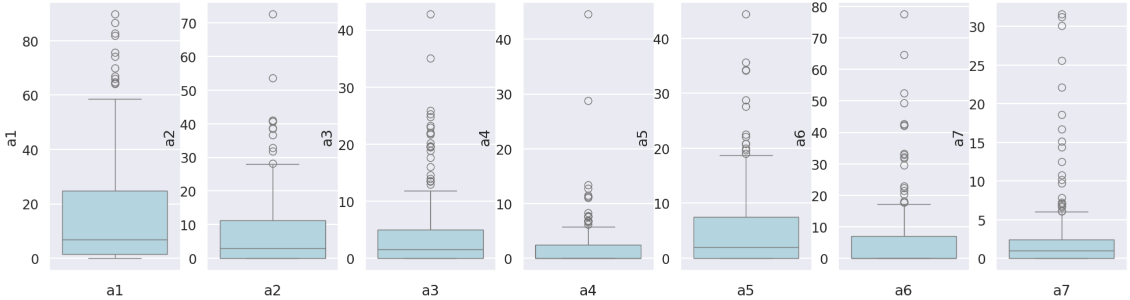
Dataset: `Algae Blooms`, Target: `a1`, . . . , `a7`

```
fig, axs = plt.subplots(1, 7, figsize=(24,6))
for k, c in enumerate(targets):
    sns.boxplot(data=df, y=c, color="lightblue", ax=axs[k])
    axs[k].set_xlabel(c)
```



Dataset: `Algae Blooms`, Target: `a1`, . . . , `a7`

```
fig, axs = plt.subplots(1, 7, figsize=(24,6))
for k, c in enumerate(targets):
    sns.boxplot(data=df, y=c, color="lightblue", ax=axs[k])
    axs[k].set_xlabel(c)
```



The outliers are likely to be true measurements, but their presence can heavily influence the model training — common strategy is to fit two models (one with the case with target outliers and one without) to assess impact of outliers.

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.
    - No missing values

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.
  - No missing values
  - No high cardinality, and reasonable balanced.

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.
4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.
  - No missing values
  - No high cardinality, and reasonable balanced.
- Eight numerical variables `max_pH`, . . . , `mean_Chlor`

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.
  - No missing values
  - No high cardinality, and reasonable balanced.
- Eight numerical variables `max_pH`, . . . , `mean_Chlor`
- Missing values present

Individual Features

To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Sneak perview

- Three categorical variables `Season`, `Size`, and `Speed`.
  - No missing values
  - No high cardinality, and reasonable balanced.
- Eight numerical variables `max_pH`, . . . , `mean_Chlor`
- Missing values present
- Some variables heavily skewed — might need to transform.

Individual Features

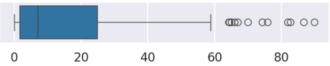
To keep this more manageable we will focus more on the `Algae Blooms` data set . . .

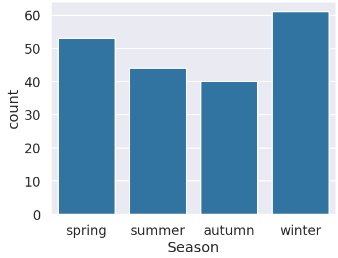
	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1	a2	a
0	winter	small	medium	8.00	9.8	60.800	6.238	578.00000	105.00000	170.00000	50.000	0.0	0.0	0.0
1	spring	small	medium	8.35	8.0	57.750	1.288	370.00000	428.75000	558.75000	1.300	1.4	7.6	4.8
2	autumn	small	medium	8.10	11.4	40.020	5.330	346.66699	125.66700	187.05701	15.600	3.3	53.6	1.9
3	spring	small	medium	8.07	4.8	77.364	2.302	98.18200	61.18200	138.70000	1.400	3.1	41.0	18.
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

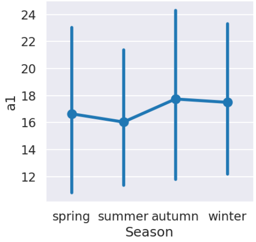
Sneak perview

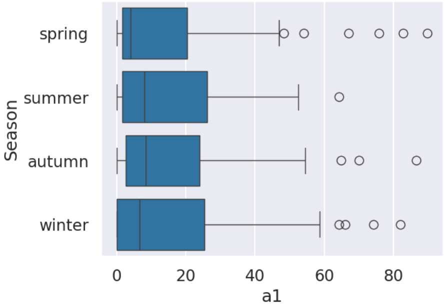
- Three categorical variables `Season`, `Size`, and `Speed`.
  - No missing values
  - No high cardinality, and reasonable balanced.
- Eight numerical variables `max_pH`, . . . , `mean_Chlor`
- Missing values present
- Some variables heavily skewed — might need to transform.
- Possibility of features being interrelated — multicollinearity — try principal component analysis.

Dataset: `Algae Blooms`, Feature: `Season`, Target: `a1`

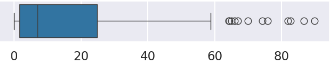


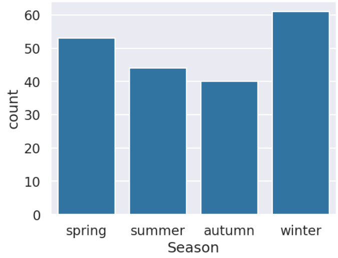


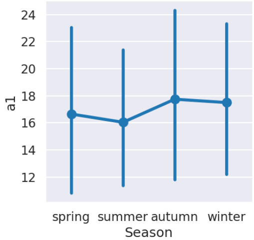


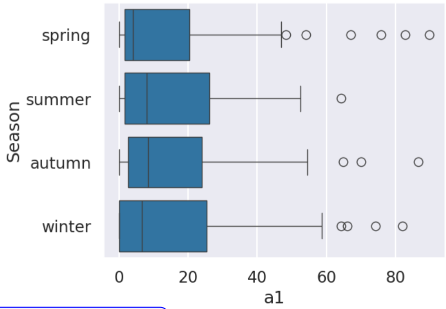


Dataset: `Algae Blooms`, Feature: `Season`, Target: `a1`





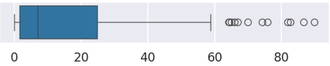


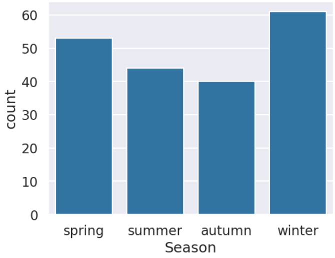


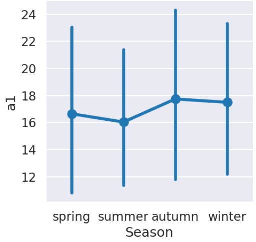

```
df.groupby("Season")["a1"].agg(["min","max","mean","count","std"])
```

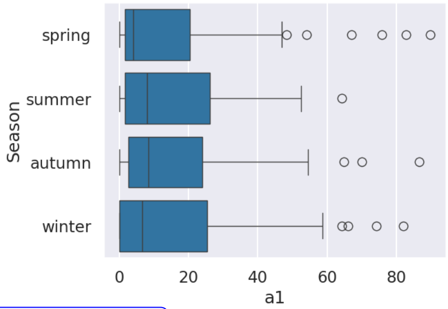
	min	max	mean	count	std
Season			$\bar{x}$	n	σ
spring	0.0	89.8	16.649057	53	23.093786
summer	0.0	64.2	16.038636	44	17.920798
autumn	0.0	86.6	17.745000	40	21.611203
winter	0.0	81.9	17.498361	61	22.568256

Dataset: `Algae Blooms`, Feature: `Season`, Target: `a1`





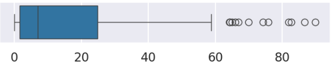




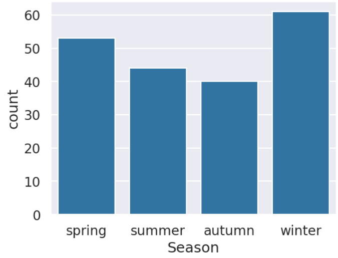

```
df.groupby("Season")["a1"].agg(["min","max","mean","count","std"])
```

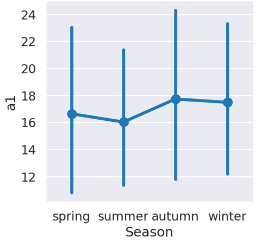
	min	max	mean	count	std
Season			$\bar{x}$	n	σ
spring	0.0	89.8	16.649057	53	23.093786
summer	0.0	64.2	16.038636	44	17.920798
autumn	0.0	86.6	17.745000	40	21.611203
winter	0.0	81.9	17.498361	61	22.568256

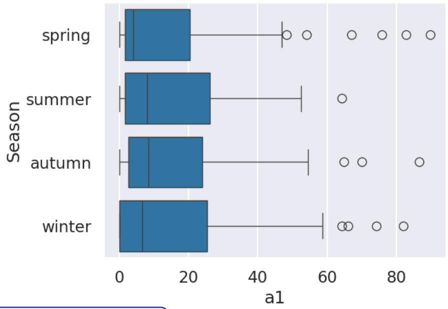
- Countplot shows no issues with feature `Season` — all levels approximately equally represented.



Dataset: `Algae Blooms`, Feature: `Season`, Target: `a1`



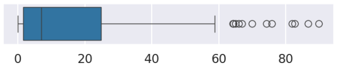




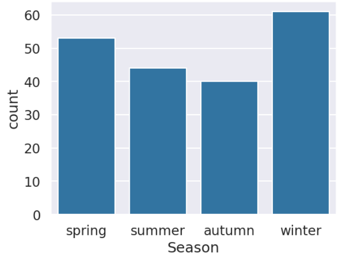

```
df.groupby("Season")["a1"].agg(["min","max","mean","count","std"])
```

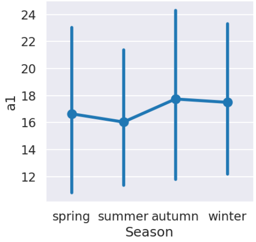
	min	max	mean	count	std
Season			$\bar{x}$	n	σ
spring	0.0	89.8	16.649057	53	23.093786
summer	0.0	64.2	16.038636	44	17.920798
autumn	0.0	86.6	17.745000	40	21.611203
winter	0.0	81.9	17.498361	61	22.568256

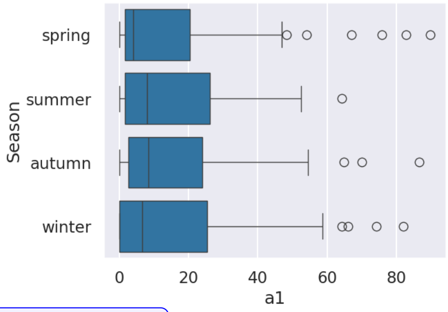
- Countplot shows no issues with feature `Season` — all levels approximately equally represented.
- Countplots show slightly less spread in `a1` for `Season="summer"` observations.



Dataset: `Algae Blooms`, Feature: `Season`, Target: `a1`





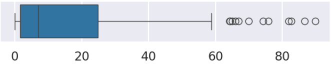


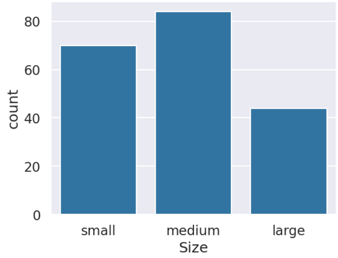

```
df.groupby("Season")["a1"].agg(["min","max","mean","count","std"])
```

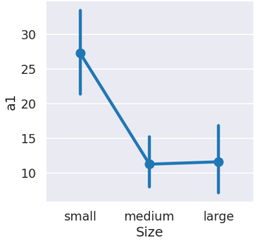
	min	max	mean	count	std
Season			$\bar{x}$	n	σ
spring	0.0	89.8	16.649057	53	23.093786
summer	0.0	64.2	16.038636	44	17.920798
autumn	0.0	86.6	17.745000	40	21.611203
winter	0.0	81.9	17.498361	61	22.568256

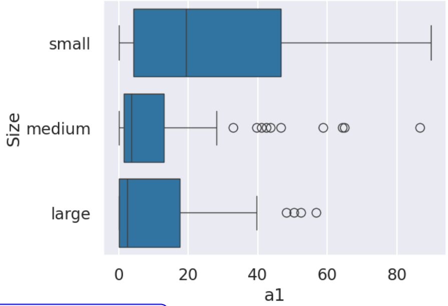
- Countplot shows no issues with feature `Season` — all levels approximately equally represented.
- Countplots show slightly less spread in `a1` for `Season="summer"` observations.
- No/weak relationship between `Season` feature and `a1` target.

Dataset: `Algae Blooms`, Feature: `Size`, Target: `a1`





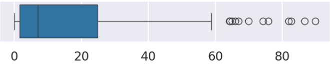


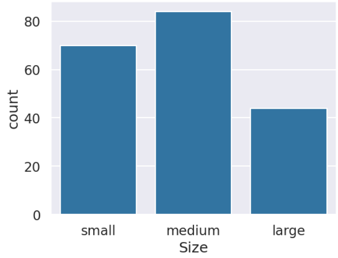


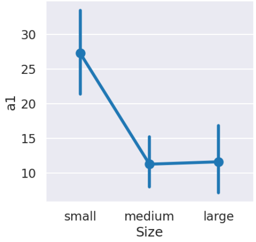

```
df.groupby("Size")["a1"].agg(["min","max","mean","count","std"])
```

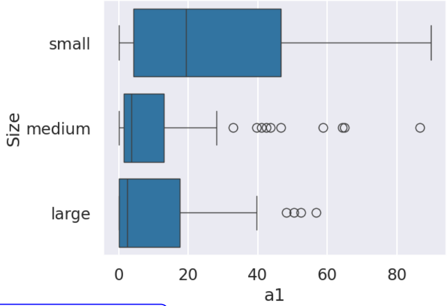
	min	max	mean	count	std
Size					
small	0.0	89.8	27.255714	70	24.895426
medium	0.0	86.6	11.267857	84	17.163124
large	0.0	56.8	11.611364	44	16.556123

Dataset: `Algae Blooms`, Feature: `Size`, Target: `a1`







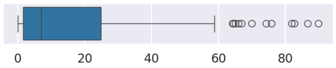


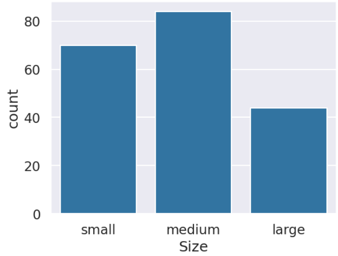

```
df.groupby("Size")["a1"].agg(["min","max","mean","count","std"])
```

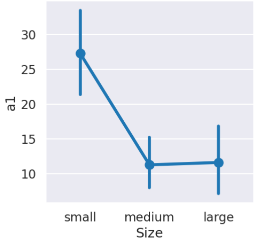
	min	max	mean	count	std
Size					
small	0.0	89.8	27.255714	70	24.895426
medium	0.0	86.6	11.267857	84	17.163124
large	0.0	56.8	11.611364	44	16.556123

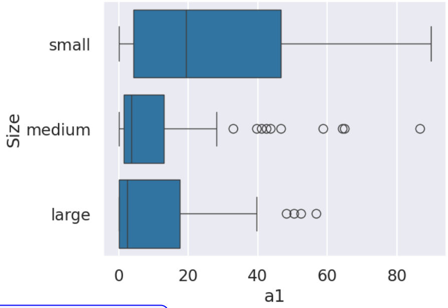
- Countplot shows no issues with feature `Size`.

Dataset: `Algae Blooms`, Feature: `Size`, Target: `a1`







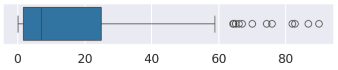


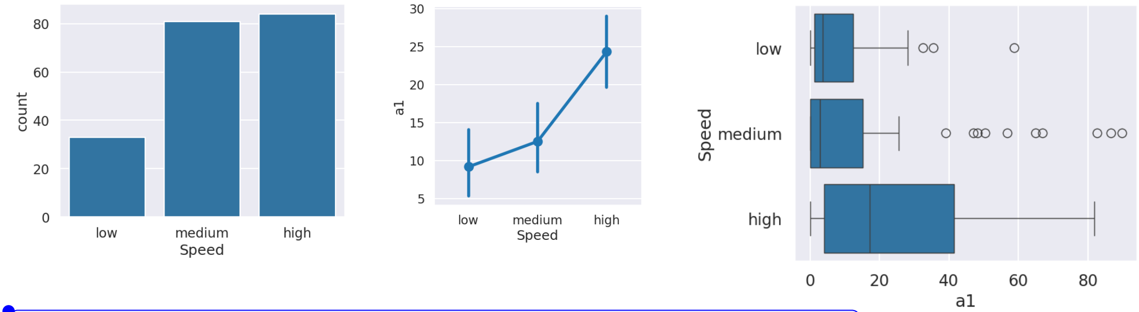

```
df.groupby("Size")["a1"].agg(["min","max","mean","count","std"])
```

	min	**max**	**mean**	**count**	**std**
Size					
small	0.0	89.8	27.255714	70	24.895426
medium	0.0	86.6	11.267857	84	17.163124
large	0.0	56.8	11.611364	44	16.556123

- Countplot shows no issues with feature `Size`.
- `Size="small"` rivers have higher frequencies of `a1` alga ((point) catplot), and observed frequencies for small rivers is much more widespread across the domain of frequencies than for other types of rivers (boxplot).

Dataset: `Algae Blooms`, Feature: `Speed`, Target: `a1`

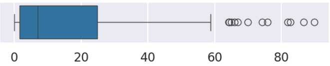


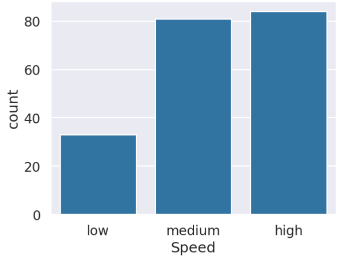


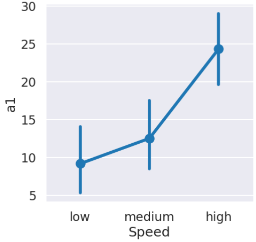

```
df.groupby("Speed", observed=False)["a1"].agg(["min","max","mean","count","std"])
```

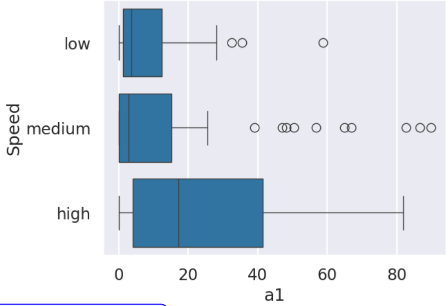
	min	max	mean	count	std
Speed					
low	0.0	58.7	9.209091	33	13.164758
medium	0.0	89.8	12.548148	81	21.146986
high	0.0	81.9	24.345238	84	22.209123

Dataset: `Algae Blooms`, Feature: `Speed`, Target: `a1`







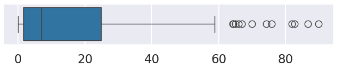


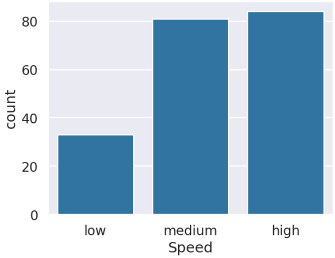

```
df.groupby("Speed", observed=False)["a1"].agg(["min","max","mean","count","std"])
```

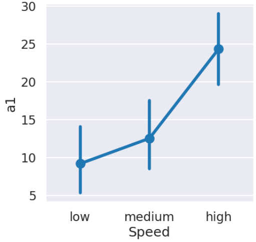
	min	max	mean	count	std
Speed					
low	0.0	58.7	9.209091	33	13.164758
medium	0.0	89.8	12.548148	81	21.146986
high	0.0	81.9	24.345238	84	22.209123

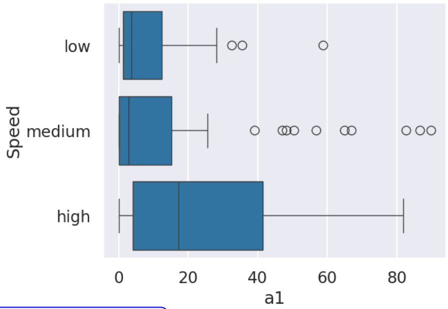
- Countplot shows no issues with feature `Speed`.

Dataset: `Algae Blooms`, Feature: `Speed`, Target: `a1`







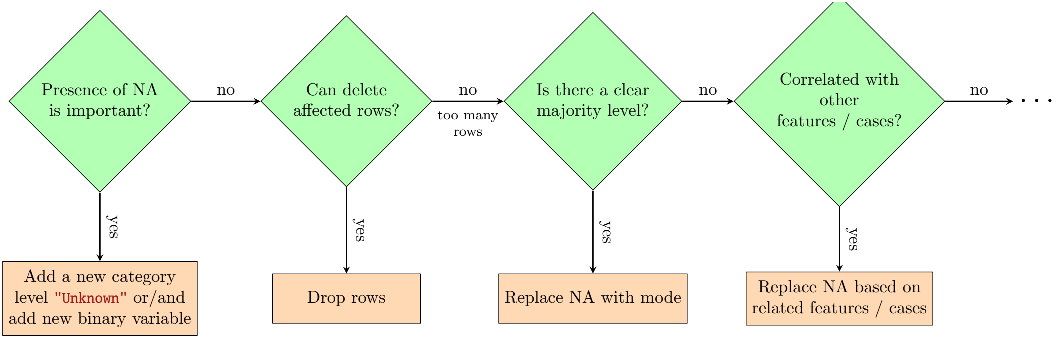



```
df.groupby("Speed", observed=False)["a1"].agg(["min","max","mean","count","std"])
```

	min	max	mean	count	std
Speed					
low	0.0	58.7	9.209091	33	13.164758
medium	0.0	89.8	12.548148	81	21.146986
high	0.0	81.9	24.345238	84	22.209123

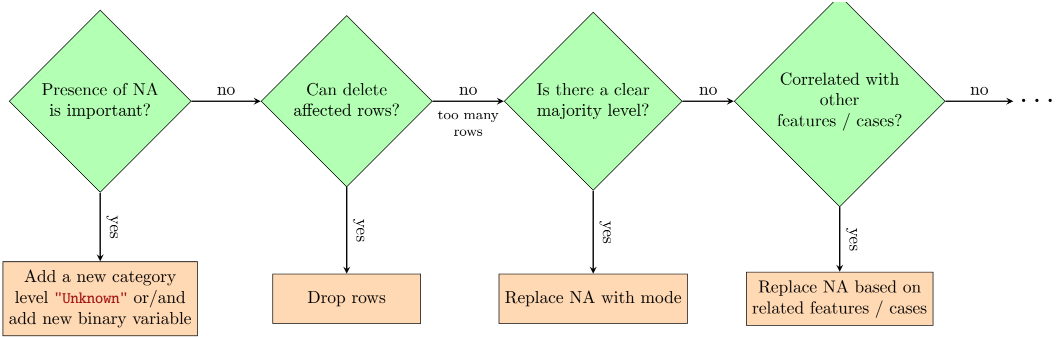
- Countplot shows no issues with feature `Speed`.
- `Speed="high"` rivers have average population of `a1` alga ((point) catplot), and observed frequencies is much more widespread across the domain of frequencies than for other types of rivers (boxplot).

Categorical Variables — Dealing with Missing Values



In terms of our three datasets, only `Titanic` has missing values in categorical features:

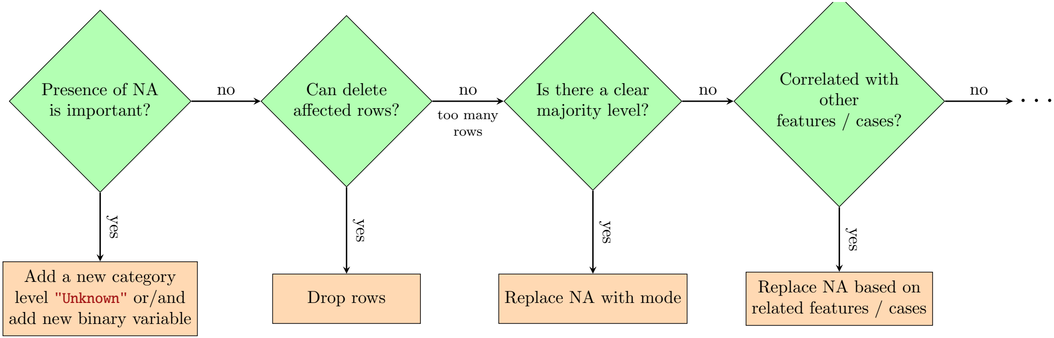
Categorical Variables — Dealing with Missing Values



In terms of our three datasets, only `Titanic` has missing values in categorical features:

- Location of `cabin`'s missing values are important (1st class passengers were most likely to have a cabin) so add new category level `"Unknown"`.

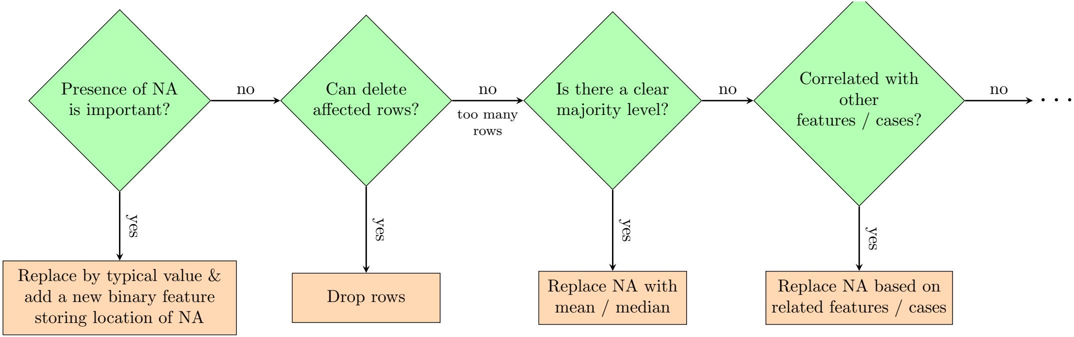
Categorical Variables — Dealing with Missing Values



In terms of our three datasets, only `Titanic` has missing values in categorical features:

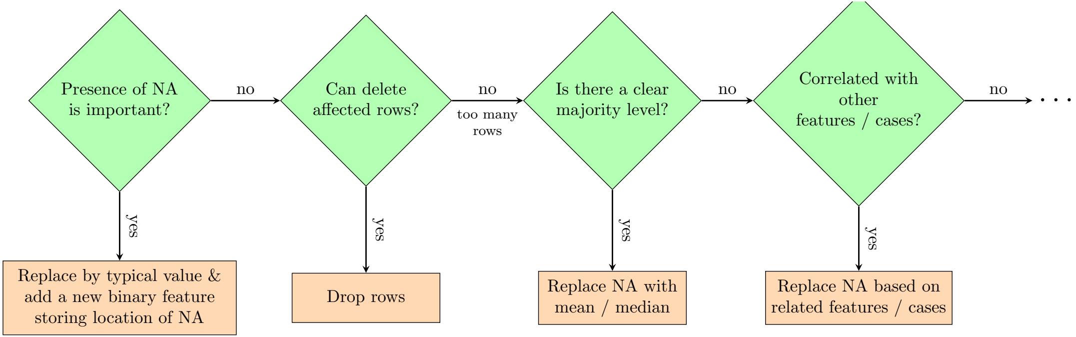
- Location of `cabin`'s missing values are important (1st class passengers were most likely to have a cabin) so add new category level `"Unknown"`.
- Replace `Embarked`'s 2 missing values with mode (`"S"`, 644/891=72%).
  Note: Use `df.Embarked.value_counts(dropna=False)` to include missing values in count tables.

Numerical Variables — Dealing with Missing Values



In terms of our three datasets:

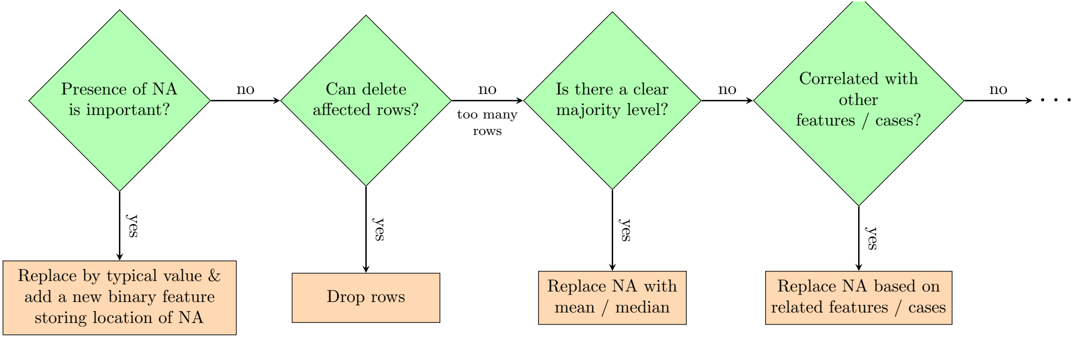
Numerical Variables — Dealing with Missing Values



In terms of our three datasets:

- In `Titanic`, feature `Fare` appears to have no missing values, but has 15 zero entries. Are these missing values? or free tickets due to age? ...

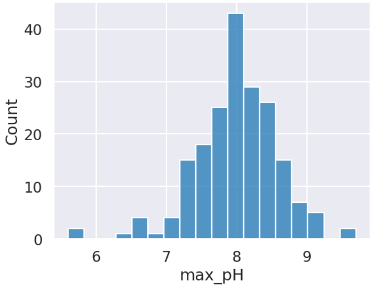
Numerical Variables — Dealing with Missing Values

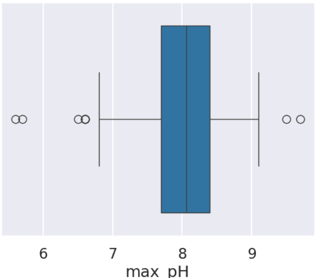


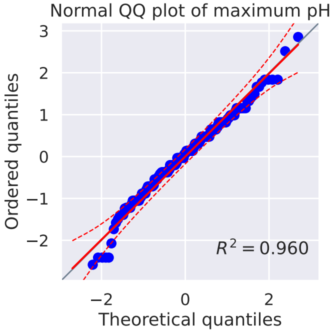
In terms of our three datasets:

- In `Titanic`, feature `Fare` appears to have no missing values, but has 15 zero entries. Are these missing values? or free tickets due to age? . . .
- In `Algae Blooms`, some of the 8 numeric features have NAs . . . next few slides.

Dataset: `Algae Blooms`, Feature: `max_pH`



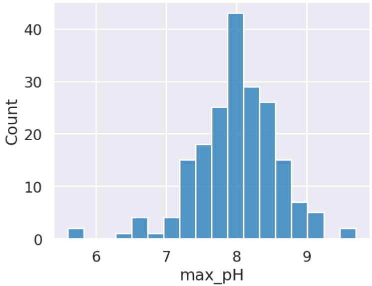


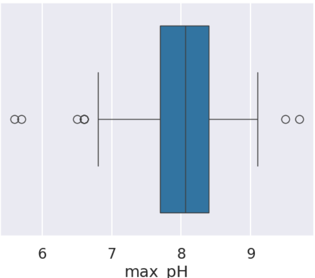


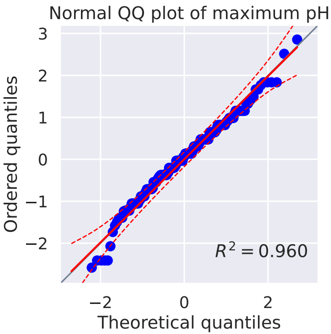

```
count    197.000000
mean       8.019975
std        0.590169
min        5.600000
25%        7.700000
50%        8.060000
75%        8.400000
max        9.700000
Name: max_pH, dtype: float64
```

- Data is relatively normal — minor issue with (left) outliers.

Dataset: `Algae Blooms`, Feature: `max_pH`







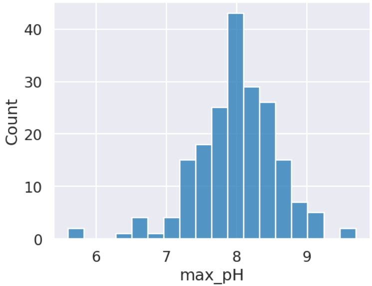

```
df.max_pH.isna().sum()
```

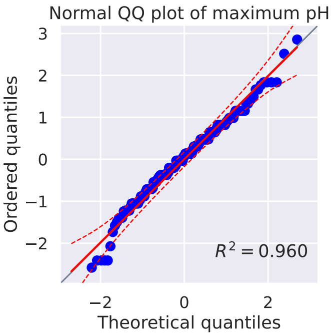
```
1
```

```
count    197.000000
mean       8.019975
std        0.590169
min        5.600000
25%        7.700000
50%        8.060000
75%        8.400000
max        9.700000
Name: max_pH, dtype: float64
```

- Data is relatively normal — minor issue with (left) outliers.

Dataset: `Algae Blooms`, Feature: `max_pH`

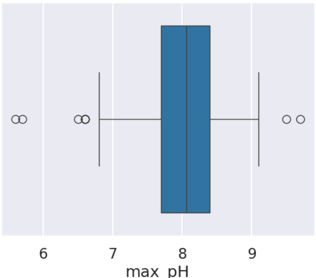





```
df.max_pH.isna().sum()
```

```
1
```

```
count    197.000000
mean       8.019975
std        0.590169
min        5.600000
25%        7.700000
50%        8.060000
75%        8.400000
max        9.700000
Name: max_pH, dtype: float64
```

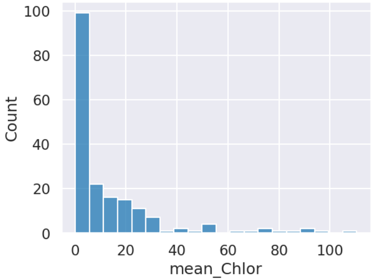


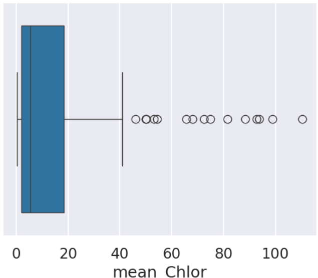
- Data is relatively normal — minor issue with (left) outliers.

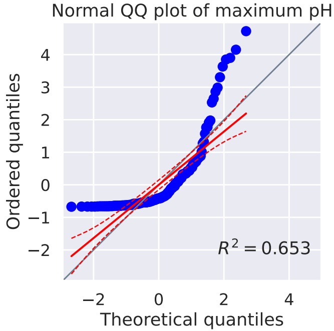
$\Rightarrow$ Will replace (single) NA by mean

```
df.max_pH = df.max_pH.fillna(df.max_pH.mean())
```

Dataset: `Algae Blooms`, Feature: `mean_Chlor`



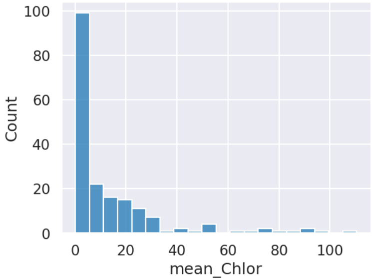


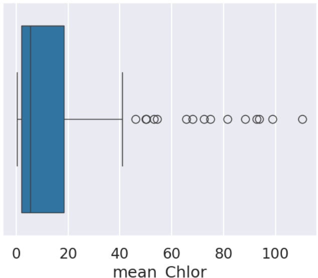


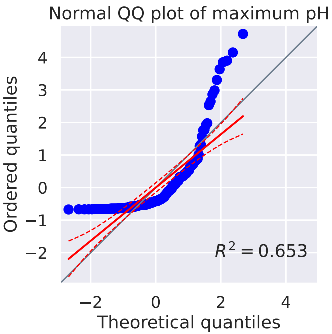

```
count    188.000000
mean      13.971197
std       20.495920
min        0.200000
25%        2.000000
50%        5.475000
75%       18.307500
max      110.456000
Name: mean_Chlor, dtype: float
```

- Data is not normal, heavily skewed to the right

Dataset: `Algae Blooms`, Feature: `mean_Chlor`







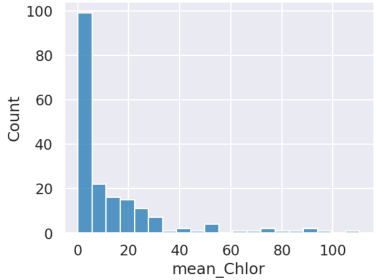

```
df.mean_Chlor.isna().sum()
```

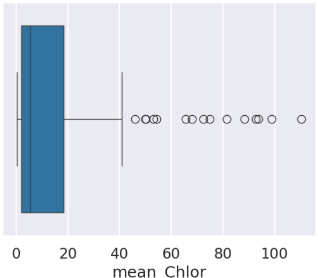
```
10
```

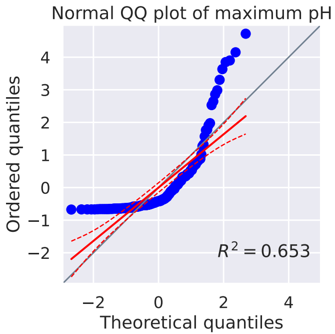
```
count     188.000000
mean       13.971197
std        20.495920
min         0.200000
25%         2.000000
50%         5.475000
75%        18.307500
max       110.456000
Name: mean_Chlor, dtype: float
```

- Data is not normal, heavily skewed to the right
- skew $\Rightarrow$ mean is a poor representative of the central location.

Dataset: `Algae Blooms`, Feature: `mean_Chlor`








```
df.mean_Chlor.isna().sum()
10
```

```
count    188.000000
mean      13.971197
std       20.495920
min        0.200000
25%        2.000000
50%        5.475000
75%       18.307500
max      110.456000
Name: mean_Chlor, dtype: float
```

- Data is not normal, heavily skewed to the right
- skew $\Rightarrow$ mean is a poor representative of the central location.
- So will replace (single) NA by median, not the mean

```
df.mean_Chlor = df.mean_Chlor.fillna(df.mean_Chlor.median()
```

After Target and Individual Feature Pass — Where are we? I

Tips

- Reviewed each feature — location, spread, shape, issues.
- No missing values
- `total_bill`, and `total_tip` have possible outliers.

After Target and Individual Feature Pass — Where are we? I

Tips

- Reviewed each feature — location, spread, shape, issues.
- No missing values
- `total_bill`, and `total_tip` have possible outliers.

Titanic

- Reviewed each feature — location, spread, shape, issues.
- Generated ToDo list for cleaning, feature extraction
  - Identified features that appear to be related to the target.
  - Feature `age` has missing values.
  - Feature `Fare`
    - has 15 measurements with value 0 — decide missing value or not.
    - distribution has large outliers and is skewed — remove/fix outliers and transform.
  - Feature `Name` has could be used to obtain new feature `Title`.
  - ...

After Target and Individual Feature Pass — Where are we? I

Tips

- Reviewed each feature — location, spread, shape, issues.
- No missing values
- `total_bill`, and `total_tip` have possible outliers.

Titanic

- Reviewed each feature — location, spread, shape, issues.
- Generated ToDo list for cleaning, feature extraction
  - Identified features that appear to be related to the target.
  - Feature `age` has missing values.
  - Feature `Fare`
    - has 15 measurements with value 0 — decide missing value or not.
    - distribution has large outliers and is skewed — remove/fix outliers and transform.
  - Feature `Name` has could be used to obtain new feature `Title`.
  - ...

Algae Blooms

- Reviewed each feature — location, spread, shape, issues.
- Imputed missing values using feature distributions (mean/median).
- Identified features that appear to be related to the target.

Aside: Steps needed to create new feature `Title` from feature `Name`

```
df = pd.read_csv('assets/train.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28		
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925		
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

1 Have a (text) feature representing the passenger's name.

Aside: Steps needed to create new feature `Title` from feature `Name`

```
df = pd.read_csv('assets/train.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28		
				Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925		
				Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
				Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

1 Have a (text) feature representing the passenger's name.

Title	Count
Capt	1
Col	4
Don	1
Dona	1
Dr	8
Jonkheer	1
Lady	1
Major	2
Master	61
Miss	260
Mlle	2
Mme	1
Mr	757
Mrs	197
Ms	2
Rev	8
Sir	1
the Countess	1

Aside: Steps needed to create new feature `Title` from feature `Name`

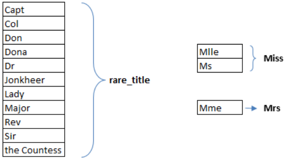
```
df = pd.read_csv('assets/train.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28		
				Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925		
				Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
				Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

1 Have a (text) feature representing the passenger's name.

Title	Count
Capt	1
Col	4
Don	1
Dona	1
Dr	8
Jonkheer	1
Lady	1
Major	2
Master	61
Miss	260
Mlle	2
Mme	1
Mr	757
Mrs	197
Ms	2
Rev	8
Sir	1
the Countess	1

2 Rare categories can be merged under a single label – 'Dona', 'Lady', 'the Countess', 'Capt', 'Col', etc. .
Duplicate/redundant categories can be merged – the titles 'Mlle', and 'Ms' and can be merged under 'Miss'. 'Mme' can be merged with 'Mrs'.



Aside: Steps needed to create new feature `Title` from feature `Name`

```
df = pd.read_csv('assets/train.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28		
				Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925		
				Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
				Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

1 Have a (text) feature representing the passenger's name.

Title	Count
Capt	1
Col	4
Don	1
Dona	1
Dr	8
Jonkheer	1
Lady	1
Major	2
Master	61
Miss	260
Mlle	2
Mme	1
Mr	757
Mrs	197
Ms	2
Rev	8
Sir	1
the Countess	1

2 Rare categories can be merged under a single label – 'Dona', 'Lady', 'the Countess', 'Capt', 'Col', etc. .
Duplicate/redundant categories can be merged – the titles 'Mlle', and 'Ms' and can be merged under 'Miss'. 'Mme' can be merged with 'Mrs'.

Capt, Col, Don, Dona, Dr, Jonkheer, Lady, Major, Rev, Sir, the Countess } rare_title

Mlle, Ms } Miss

Mme → Mrs

3 New categorical feature with 5 unique values

Master	Miss	Mr	Mrs	rare_title
61	264	757	198	29

Third Pass — Relationships Between Features (and Target)

- Correlations

We distinguish later between feature-feature and feature-target correlations

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

	Feature
Categorical	`nunique`, `unique`, `describe`, `value_counts`, ...
	`countplot`, ...

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

	Feature
Categorical	`nunique`, `unique`, `describe`, `value_counts`, ...
	`countplot`, ...
Numerical	`describe`, ...
	`histplot`, `boxplot`, `displot`, `qqplot`, ...

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

		Relationship to Target	
	Feature	Categorical	Numerical
Categorical	`nunique`, `unique`, `describe`, `value_counts`, ...		
	`countplot`, ...		
Numerical	`describe`, ...		
	`histplot`, `boxplot`, `displot`, `qqplot`, ...		

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

		Relationship to Target	
	Feature	Categorical	Numerical
Categorical	`nunique`, `unique`, `describe`, `value_counts`, … `countplot`, …	`crosstab`, … `countplot`, …	
Numerical	`describe`, … `histplot`, `boxplot`, `displot`, `qqplot`, …		

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

	Feature	Relationship to Target: Categorical	Relationship to Target: Numerical
Categorical	`nunique`, `unique`, `describe`, `value_counts`, … `countplot`, …	`crosstab`, … `countplot`, …	
Numerical	`describe`, … `histplot`, `boxplot`, `displot`, `qqplot`, …	`groupby+describe`, … `catplot`, `boxplot`, …	

Categorical and Numerical variables

A Selection of Statistical Visualisations and Metrics

		Relationship to Target	
	Feature	Categorical	Numerical
Categorical	`nunique`, `unique`, `describe`, `value_counts`, … `countplot`, …	`crosstab`, … `countplot`, …	`boxplot`, … `catplot`, `boxplot`, …
Numerical	`describe`, … `histplot`, `boxplot`, `displot`, `qqplot`, …	`groupby+describe`, … `catplot`, `boxplot`, …	

Categorical and Numerical variables

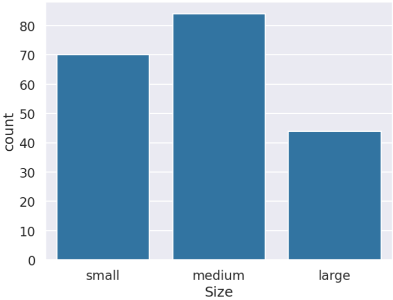
A Selection of Statistical Visualisations and Metrics

Relationship to Target

	Feature	Categorical	Numerical
Categorical	nunique, unique, describe, value_counts, …	crosstab, …	boxplot, …
	countplot, …	countplot, …	catplot, boxplot, …
Numerical	describe, …	groupby+describe, …	correlations, …
	histplot, boxplot, displot, qqplot, …	catplot, boxplot, …	lmplot, …

Categorical Variables — Relationship with (Numerical) Target I

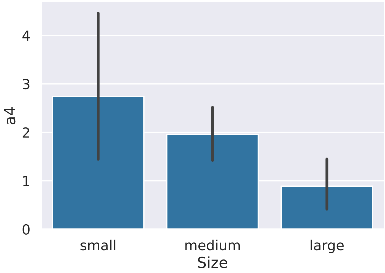
```
sns.countplot(x="Size", data=df);
```



- Shows the counts of observations in each categorical level using bar (height/width).

Is it usable?

```
sns.catplot(x="Size", y="a4", data=df, kind='bar');
```

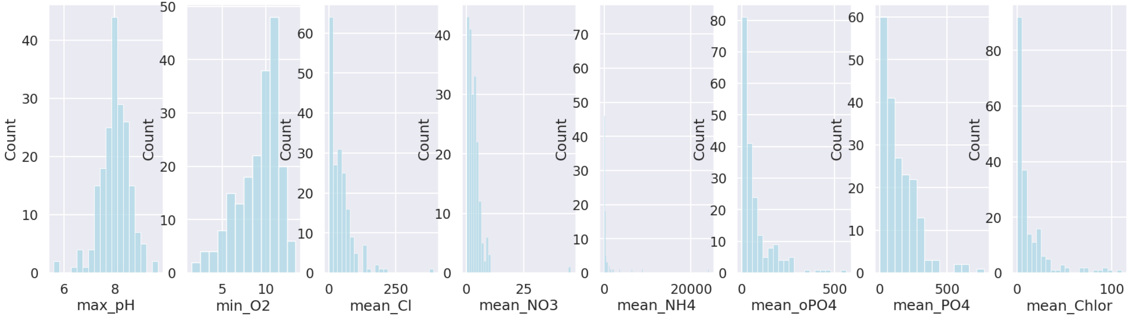


- Shows the average level (mean) and uncertainty (std) of the numerical target (`a4`) in each categorical level of the categorical variable.
- Vertical bar shows 95% confidence interval.

Is it useful?

Example — Dataset: `Algae_Blooms`, Feature: all numeric I

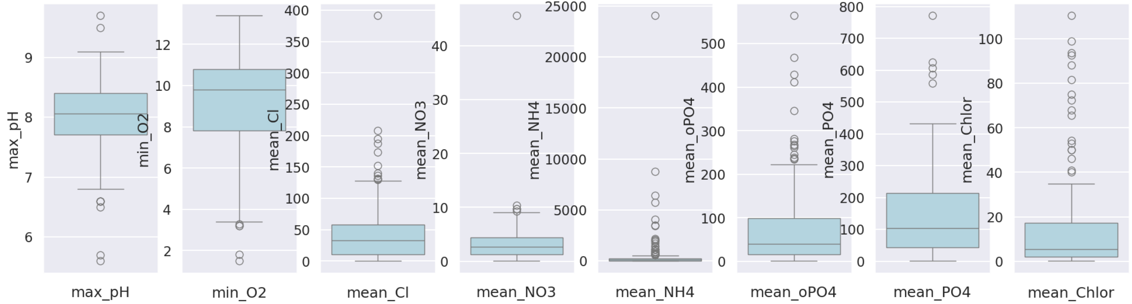
```
fig, axs = plt.subplots(1, 8, figsize=(24,6))
for k, c in enumerate(df.columns[3:11]):
    sns.histplot(data=df, x=c, color="lightblue", ax=axs[k])
    axs[k].set_xlabel(c)
```



- Most features are heavily skewed, `max_pH` appears to be least skewed.

Example — Dataset: `Algae_Blooms`, Feature: all numeric II

```python
fig, axs = plt.subplots(1, 8, figsize=(24,6))
for k, c in enumerate(df.columns[3:11]):
    sns.boxplot(data=df, y=c, color="lightblue", ax=axs[k])
    axs[k].set_xlabel(c)
```



- Outliers are much clearer. `mean_NH4`, `mean_NO3`, and `meanCL` have clear outliers that need to be addressed.
- On the other hand features like `mean_Chlor` and `mean_oPO4` appear to follow a skewed distribution and need to be transformed.

Dataset: `Algae Blooms`, Feature: `max_pH`, Target: `a1`

Is there a relationship between feature `max_pH` and target `a1`?

Dataset: `Algae Blooms`, Feature: `max_pH`, Target: `a1`

Is there a relationship between feature `max_pH` and target `a1`?

```
df[["max_pH", "a1"]].corr()
```

	max_pH	a1
max_pH	1.000000	-0.268539
a1	-0.268539	1.000000

Dataset: Algae Blooms, Feature: max_pH, Target: a1

Is there a relationship between feature `max_pH` and target `a1`?

```
df[["max_pH", "a1"]].corr()
```

	max_pH	a1
max_pH	1.000000	-0.268539
a1	-0.268539	1.000000

(Pearson's) Correlation coefficient, r, measures the strength of a **linear** relationship between two numerical variables.

- near zero means no/weak linear relationship.
- near ± 1 zero means strong linear relationship.
- sign indicates direction of relationship

Dataset: `Algae Blooms`, Feature: `max_pH`, Target: `a1`

Is there a relationship between feature `max_pH` and target `a1`?

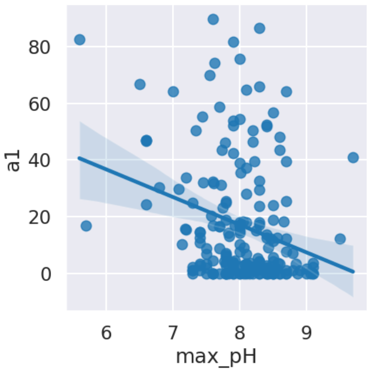
```
df[["max_pH", "a1"]].corr()
```

	max_pH	**a1**
max_pH	1.000000	-0.268539
a1	-0.268539	1.000000

(Pearson's) Correlation coefficient, r, measures the strength of a **linear** relationship between two numerical variables.

- near zero means no/weak linear relationship.
- near ± 1 zero means strong linear relationship.
- sign indicates direction of relationship

```
sns.lmplot(x="max_pH", y="a1", data=df);
```



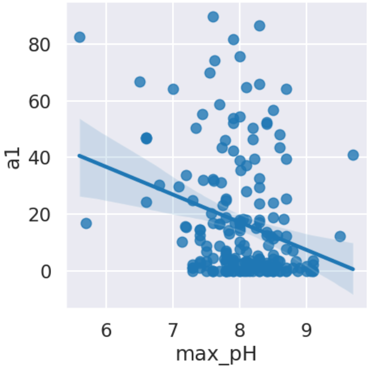
Dataset: `Algae Blooms`, Feature: `max_pH`, Target: `a1`

Is there a relationship between feature `max_pH` and target `a1`?

```
df[["max_pH", "a1"]].corr()
```

```
sns.lmplot(x="max_pH", y="a1", data=df);
```

	max_pH	**a1**
max_pH	1.000000	-0.268539
a1	-0.268539	1.000000



(Pearson's) Correlation coefficient, r, measures the strength of a **linear** relationship between two numerical variables.

- near zero means no/weak linear relationship.
- near ± 1 zero means strong linear relationship.
- sign indicates direction of relationship

- Correlation coefficient, $r = -0.27$, shows (at most) a weak negative linear relationship.
- No obvious relationship visible in scatter plot.

Correlations — Relationship Between two Variables

Pearson's correlation coefficient, r

is a measure of linear correlation between two variables. Its value lies between -1 and +1, -1 indicating total negative linear correlation, 0 indicating no linear correlation and 1 indicating total positive linear correlation.

Correlations — Relationship Between two Variables

Pearson's correlation coefficient, r

is a measure of linear correlation between two variables. Its value lies between -1 and +1, -1 indicating total negative linear correlation, 0 indicating no linear correlation and 1 indicating total positive linear correlation.

Spearman's rank correlation coefficient, ρ

is a measure of monotonic correlation between two variables, and is therefore better in catching nonlinear monotonic correlations than Pearson's r. Its value also lies between -1 and +1, with values near zero indicating no monotonic relation.

Correlations — Relationship Between two Variables

Pearson's correlation coefficient, r

is a measure of linear correlation between two variables. Its value lies between -1 and +1, -1 indicating total negative linear correlation, 0 indicating no linear correlation and 1 indicating total positive linear correlation.

Spearman's rank correlation coefficient, ρ

is a measure of monotonic correlation between two variables, and is therefore better in catching nonlinear monotonic correlations than Pearson's r. Its value also lies between -1 and +1, with values near zero indicating no monotonic relation.

Kendall rank correlation coefficient, τ

measures ordinal association between two variables. Its value lies between -1 and +1 with values near zero indicating no relation.

Correlations — Relationship Between two Variables

Pearson's correlation coefficient, r

is a measure of linear correlation between two variables. Its value lies between -1 and +1, -1 indicating total negative linear correlation, 0 indicating no linear correlation and 1 indicating total positive linear correlation.

Spearman's rank correlation coefficient, ρ

is a measure of monotonic correlation between two variables, and is therefore better in catching nonlinear monotonic correlations than Pearson's r. Its value also lies between -1 and +1, with values near zero indicating no monotonic relation.

Kendall rank correlation coefficient, τ

measures ordinal association between two variables. Its value lies between -1 and +1 with values near zero indicating no relation.

Phi-k, ϕk

is a new and practical correlation coefficient that works consistently between categorical, ordinal and interval variables, captures non-linear dependency and reverts to the Pearson correlation coefficient in case of a bivariate normal input distribution. Its value also lies between 0 and +1, with values near zero indicating no relation.

Pearson's Correlation Coefficient — Dataset: `Algae Blooms`

```
columns = df.columns[:12]
corr = df[columns].corr(numeric_only=True)
corr
```

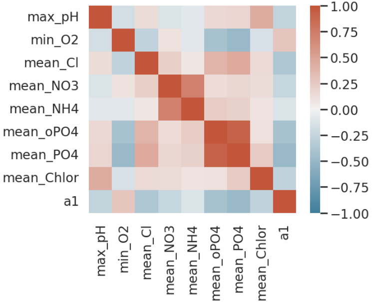
	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.167981	0.136369	-0.130762	-0.093521	0.158769	0.179885	0.445864	-0.268539
min_O2	-0.167981	1.000000	-0.278333	0.099444	-0.087478	-0.416163	-0.487486	-0.153265	0.285564
mean_Cl	0.136369	-0.278333	1.000000	0.225041	0.071913	0.391054	0.457449	0.149856	-0.371171
mean_NO3	-0.130762	0.099444	0.225041	1.000000	0.721444	0.144588	0.168601	0.139679	-0.241211
mean_NH4	-0.093521	-0.087478	0.071913	0.721444	1.000000	0.227237	0.208180	0.088947	-0.132656
mean_oPO4	0.158769	-0.416163	0.391054	0.144588	0.227237	1.000000	0.914365	0.115621	-0.417358
mean_PO4	0.179885	-0.487486	0.457449	0.168601	0.208180	0.914365	1.000000	0.253621	-0.487023
mean_Chlor	0.445864	-0.153265	0.149856	0.139679	0.088947	0.115621	0.253621	1.000000	-0.277987
a1	-0.268539	0.285564	-0.371171	-0.241211	-0.132656	-0.417358	-0.487023	-0.277987	1.000000

Pearson's Correlation Coefficient — Dataset: `Algae Blooms`

```
columns = df.columns[:12]
corr = df[columns].corr(numeric_o
corr
```

```
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(corr, square=True, vmin=-1, vmax=1, cmap=cmap);
```

	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.167981	0.136369	-0.130762	-0.093521	0.158769	0.179885	0.445864	-0.268539
min_O2	-0.167981	1.000000	-0.278333	0.099444	-0.087478	-0.416163	-0.487486	-0.153265	0.285564
mean_Cl	0.136369	-0.278333	1.000000	0.225041	0.071913	0.391054	0.457449	0.149856	-0.371171
mean_NO3	-0.130762	0.099444	0.225041	1.000000	0.721444	0.144588	0.168601	0.139679	-0.241211
mean_NH4	-0.093521	-0.087478	0.071913	0.721444	1.000000	0.227237	0.208180	0.088947	-0.132656
mean_oPO4	0.158769	-0.416163	0.391054	0.144588	0.227237	1.000000	0.914365	0.115621	-0.417358
mean_PO4	0.179885	-0.487486	0.457449	0.168601	0.208180	0.914365	1.000000	0.253621	-0.487023
mean_Chlor	0.445864	-0.153265	0.149856	0.139679	0.088947	0.115621	0.253621	1.000000	-0.277987
a1	-0.268539	0.285564	-0.371171	-0.241211	-0.132656	-0.417358	-0.487023	-0.277987	1.000000

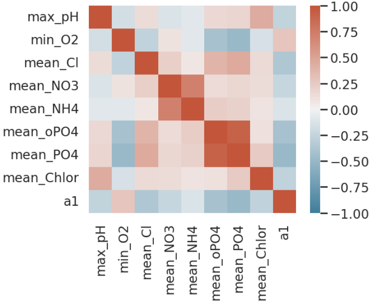


Pearson's Correlation Coefficient — Dataset: `Algae Blooms`

```
columns = df.columns[:12]
corr = df[columns].corr(numeric_o
corr
```

```
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(corr, square=True, vmin=-1, vmax=1, cmap=cmap);
```

	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.167981	0.136369	-0.130762	-0.093521	0.158769	0.179885	0.445864	-0.268539
min_O2	-0.167981	1.000000	-0.278333	0.099444	-0.087478	-0.416163	-0.487486	-0.153265	0.285564
mean_Cl	0.136369	-0.278333	1.000000	0.225041	0.071913	0.391054	0.457449	0.149856	-0.371171
mean_NO3	-0.130762	0.099444	0.225041	1.000000	0.721444	0.144588	0.168601	0.139679	-0.241211
mean_NH4	-0.093521	-0.087478	0.071913	0.721444	1.000000	0.227237	0.208180	0.088947	-0.132656
mean_oPO4	0.158769	-0.416163	0.391054	0.144588	0.227237	1.000000	0.914365	0.115621	-0.417358
mean_PO4	0.179885	-0.487486	0.457449	0.168601	0.208180	0.914365	1.000000	0.253621	-0.487023
mean_Chlor	0.445864	-0.153265	0.149856	0.139679	0.088947	0.115621	0.253621	1.000000	-0.277987
a1	-0.268539	0.285564	-0.371171	-0.241211	-0.132656	-0.417358	-0.487023	-0.277987	1.000000



- Categorical variables are not included.
- Suggests best predictors for `a1` are `mean_PO4`, `mean_oPO4`, and `meanCl`.
- `mean_PO4` and `mean_oPO4` are highly correlated (0.91) — could use values of one to estimate missing values of the other.

Spearman's Rank Correlation Coefficient — Dataset: Algae Blooms

```
columns = df.columns[:12]
corr = df[columns].corr(method='spearman', numeric_only=True)
corr
```

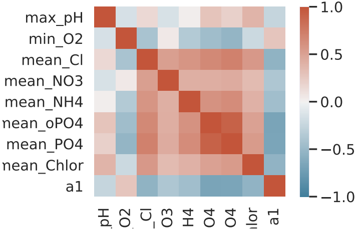
	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.148676	0.159079	-0.145182	0.026160	0.290245	0.214569	0.394813	-0.247787
min_O2	-0.148676	1.000000	-0.405142	0.057610	-0.348226	-0.457805	-0.519786	-0.217714	0.283418
mean_Cl	0.159079	-0.405142	1.000000	0.530374	0.592052	0.670399	0.713479	0.564915	-0.546845
mean_NO3	-0.145182	0.057610	0.530374	1.000000	0.425010	0.432303	0.451272	0.346805	-0.382403
mean_NH4	0.026160	-0.348226	0.592052	0.425010	1.000000	0.603157	0.646690	0.406656	-0.449194
mean_oPO4	0.290245	-0.457805	0.670399	0.432303	0.603157	1.000000	0.914921	0.510930	-0.671019
mean_PO4	0.214569	-0.519786	0.713479	0.451272	0.646690	0.914921	1.000000	0.554167	-0.656670
mean_Chlor	0.394813	-0.217714	0.564915	0.346805	0.406656	0.510930	0.554167	1.000000	-0.537823
a1	-0.247787	0.283418	-0.546845	-0.382403	-0.449194	-0.671019	-0.656670	-0.537823	1.000000

Spearman's Rank Correlation Coefficient — Dataset: `Algae Blooms`

```python
columns = df.columns[:12]
corr = df[columns].corr(method='spearman', nume
corr
```

```python
cmap = sns.diverging_palette(230, 20, as_cmap=Tr
sns.heatmap(corr, square=True, vmin=-1, vmax=1,
```

	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.148676	0.159079	-0.145182	0.026160	0.290245	0.214569	0.394813	-0.247787
min_O2	-0.148676	1.000000	-0.405142	0.057610	-0.348226	-0.457805	-0.519786	-0.217714	0.283418
mean_Cl	0.159079	-0.405142	1.000000	0.530374	0.592052	0.670399	0.713479	0.564915	-0.546845
mean_NO3	-0.145182	0.057610	0.530374	1.000000	0.425010	0.432303	0.451272	0.346805	-0.382403
mean_NH4	0.026160	-0.348226	0.592052	0.425010	1.000000	0.603157	0.646690	0.406656	-0.449194
mean_oPO4	0.290245	-0.457805	0.670399	0.432303	0.603157	1.000000	0.914921	0.510930	-0.671019
mean_PO4	0.214569	-0.519786	0.713479	0.451272	0.646690	0.914921	1.000000	0.554167	-0.656670
mean_Chlor	0.394813	-0.217714	0.564915	0.346805	0.406656	0.510930	0.554167	1.000000	-0.537823
a1	-0.247787	0.283418	-0.546845	-0.382403	-0.449194	-0.671019	-0.656670	-0.537823	1.000000

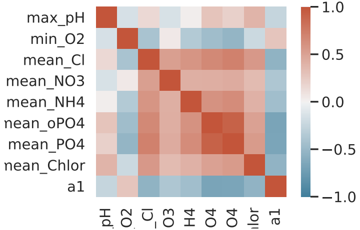


Spearman's Rank Correlation Coefficient — Dataset: `Algae Blooms`

```
columns = df.columns[:12]
corr = df[columns].corr(method='spearman', nume
corr
```

```
cmap = sns.diverging_palette(230, 20, as_cmap=Tr
sns.heatmap(corr, square=True, vmin=-1, vmax=1,
```

	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_Chlor	a1
max_pH	1.000000	-0.148676	0.159079	-0.145182	0.026160	0.290245	0.214569	0.394813	-0.247787
min_O2	-0.148676	1.000000	-0.405142	0.057610	-0.348226	-0.457805	-0.519786	-0.217714	0.283418
mean_Cl	0.159079	-0.405142	1.000000	0.530374	0.592052	0.670399	0.713479	0.564915	-0.546845
mean_NO3	-0.145182	0.057610	0.530374	1.000000	0.425010	0.432303	0.451272	0.346805	-0.382403
mean_NH4	0.026160	-0.348226	0.592052	0.425010	1.000000	0.603157	0.646690	0.406656	-0.449194
mean_oPO4	0.290245	-0.457805	0.670399	0.432303	0.603157	1.000000	0.914921	0.510930	-0.671019
mean_PO4	0.214569	-0.519786	0.713479	0.451272	0.646690	0.914921	1.000000	0.554167	-0.656670
mean_Chlor	0.394813	-0.217714	0.564915	0.346805	0.406656	0.510930	0.554167	1.000000	-0.537823
a1	-0.247787	0.283418	-0.546845	-0.382403	-0.449194	-0.671019	-0.656670	-0.537823	1.000000



- Now best predictors for `a1` also include `mean_Chlor` and `mean_NH4`.

Phik Correlation Coefficient — Dataset: `Algae Blooms`

```python
import phik
columns = df.columns[:12]
corr = df[columns].phik_matrix()
corr
```

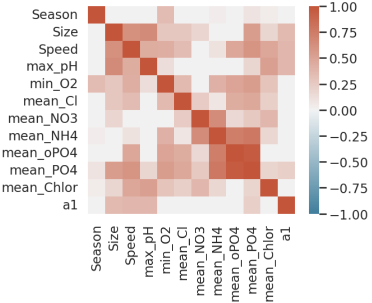
	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_C
Season	1.000000	0.000000	0.000000	0.000000	0.343496	0.000000	0.000000	0.034202	0.000000	0.093199	0.04536
Size	0.000000	1.000000	0.620101	0.655207	0.270013	0.268198	0.182410	0.000000	0.000000	0.531635	0.17351
Speed	0.000000	0.620101	1.000000	0.445096	0.437356	0.339237	0.000000	0.101348	0.483298	0.594480	0.47973
max_pH	0.000000	0.655207	0.445096	1.000000	0.125231	0.000000	0.000000	0.000000	0.000000	0.175105	0.52813
min_O2	0.343496	0.270013	0.437356	0.125231	1.000000	0.353196	0.000000	0.416999	0.492457	0.535996	0.29637
mean_Cl	0.000000	0.268198	0.339237	0.000000	0.353196	1.000000	0.243887	0.073692	0.443047	0.472824	0.22558
mean_NO3	0.000000	0.182410	0.000000	0.000000	0.000000	0.243887	1.000000	0.642789	0.158463	0.259915	0.36814
mean_NH4	0.034202	0.000000	0.101348	0.000000	0.416999	0.073692	0.642789	1.000000	0.734681	0.776197	0.16753
mean_oPO4	0.000000	0.000000	0.483298	0.000000	0.492457	0.443047	0.158463	0.734681	1.000000	0.954601	0.00000
mean_PO4	0.093199	0.531635	0.594480	0.175105	0.535996	0.472824	0.259915	0.776197	0.954601	1.000000	0.19292
mean_Chlor	0.045361	0.173516	0.479735	0.528134	0.296376	0.225583	0.368142	0.167533	0.000000	0.192920	1.00000
a1	0.000000	0.353390	0.369374	0.372031	0.000000	0.000000	0.000000	0.000000	0.000000	0.221308	0.00000

Phik Correlation Coefficient — Dataset: `Algae Blooms`

```python
import phik
columns = df.columns[:12]
corr = df[columns].phik_matrix()
corr
```

```python
cmap = sns.diverging_palette(230, 20, as_cmap=Tr
sns.heatmap(corr, square=True, vmin=-1, vmax=1,
```

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_C
Season	1.000000	0.000000	0.000000	0.000000	0.343496	0.000000	0.000000	0.034202	0.000000	0.093199	0.04536
Size	0.000000	1.000000	0.620101	0.655207	0.270013	0.268198	0.182410	0.000000	0.000000	0.531635	0.17351
Speed	0.000000	0.620101	1.000000	0.445096	0.437356	0.339237	0.000000	0.101348	0.483298	0.594480	0.47973
max_pH	0.000000	0.655207	0.445096	1.000000	0.125231	0.000000	0.000000	0.000000	0.000000	0.175105	0.52813
min_O2	0.343496	0.270013	0.437356	0.125231	1.000000	0.353196	0.000000	0.416999	0.492457	0.535996	0.29637
mean_Cl	0.000000	0.268198	0.339237	0.000000	0.353196	1.000000	0.243887	0.073692	0.443047	0.472824	0.22558
mean_NO3	0.000000	0.182410	0.000000	0.000000	0.000000	0.243887	1.000000	0.642789	0.158463	0.259915	0.36814
mean_NH4	0.034202	0.000000	0.101348	0.000000	0.416999	0.073692	0.642789	1.000000	0.734681	0.776197	0.16753
mean_oPO4	0.000000	0.000000	0.483298	0.000000	0.492457	0.443047	0.158463	0.734681	1.000000	0.954601	0.00000
mean_PO4	0.093199	0.531635	0.594480	0.175105	0.535996	0.472824	0.259915	0.776197	0.954601	1.000000	0.19292
mean_Chlor	0.045361	0.173516	0.479735	0.528134	0.296376	0.225583	0.368142	0.167533	0.000000	0.192920	1.00000
a1	0.000000	0.353390	0.369374	0.372031	0.000000	0.000000	0.000000	0.000000	0.000000	0.221308	0.00000

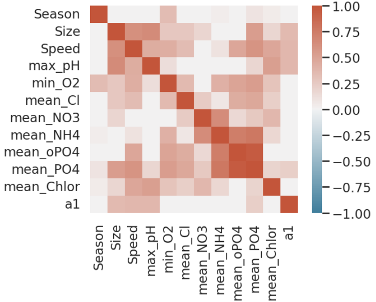


Phik Correlation Coefficient — Dataset: `Algae Blooms`

```python
import phik
columns = df.columns[:12]
corr = df[columns].phik_matrix()
corr
```

```python
cmap = sns.diverging_palette(230, 20, as_cmap=Tr
sns.heatmap(corr, square=True, vmin=-1, vmax=1,
```

	Season	Size	Speed	max_pH	min_O2	mean_Cl	mean_NO3	mean_NH4	mean_oPO4	mean_PO4	mean_C
Season	1.000000	0.000000	0.000000	0.000000	0.343496	0.000000	0.000000	0.034202	0.000000	0.093199	0.04536
Size	0.000000	1.000000	0.620101	0.655207	0.270013	0.268198	0.182410	0.000000	0.000000	0.531635	0.17351
Speed	0.000000	0.620101	1.000000	0.445096	0.437356	0.339237	0.000000	0.101348	0.483298	0.594480	0.47973
max_pH	0.000000	0.655207	0.445096	1.000000	0.125231	0.000000	0.000000	0.000000	0.000000	0.175105	0.52813
min_O2	0.343496	0.270013	0.437356	0.125231	1.000000	0.353196	0.000000	0.416999	0.492457	0.535996	0.29637
mean_Cl	0.000000	0.268198	0.339237	0.000000	0.353196	1.000000	0.243887	0.073692	0.443047	0.472824	0.22558
mean_NO3	0.000000	0.182410	0.000000	0.000000	0.000000	0.243887	1.000000	0.642789	0.158463	0.259915	0.36814
mean_NH4	0.034202	0.000000	0.101348	0.000000	0.416999	0.073692	0.642789	1.000000	0.734681	0.776197	0.16753
mean_oPO4	0.000000	0.000000	0.483298	0.000000	0.492457	0.443047	0.158463	0.734681	1.000000	0.954601	0.00000
mean_PO4	0.093199	0.531635	0.594480	0.175105	0.535996	0.472824	0.259915	0.776197	0.954601	1.000000	0.19292
mean_Chlor	0.045361	0.173516	0.479735	0.528134	0.296376	0.225583	0.368142	0.167533	0.000000	0.192920	1.00000
a1	0.000000	0.353390	0.369374	0.372031	0.000000	0.000000	0.000000	0.000000	0.000000	0.221308	0.00000

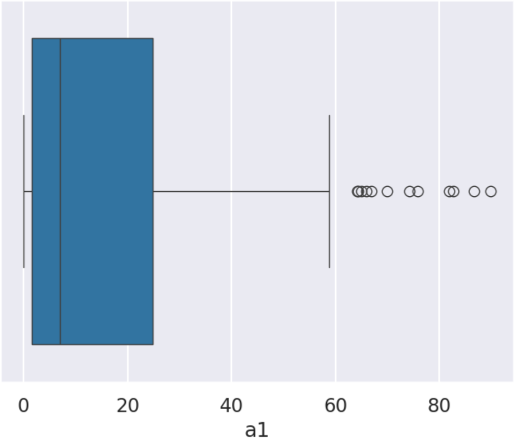


- Now include categorical variables — `Season` is not related, but `Size` and `Speed` are.

Multi-Relation Plots

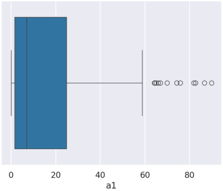
```
sns.boxplot(x="a1", data=df);
```

•

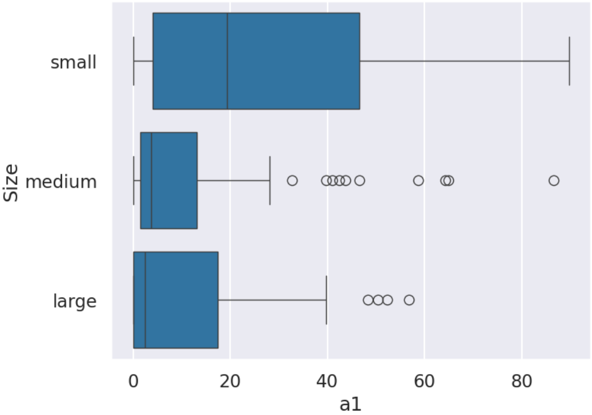


Multi-Relation Plots

```
sns.boxplot(x="a1", data=df);
```




```
sns.boxplot(x="a1", y="Size", data=df);
```

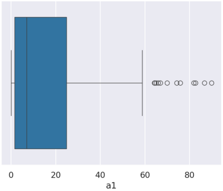


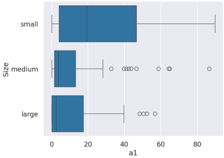
Multi-Relation Plots

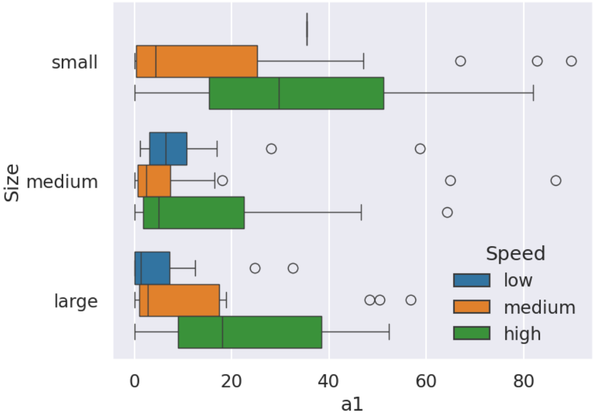
```
sns.boxplot(x="a1", data=df);
```

```
sns.boxplot(x="a1", y="Size", data=df);
```

```
sns.boxplot(x="a1", y="Size", hue="Speed", data=df);
```







- If have n features, then have $n(n-1)$ possible visualisations — gets a bit crazy.

After Third Pass — Where are we?

- Reviewed each feature — location, spread, shape, issues.

After Third Pass — Where are we?

- Reviewed each feature — location, spread, shape, issues.
- Identified any correlation among features and with target.

After Third Pass — Where are we?

- Reviewed each feature — location, spread, shape, issues.
- Identified any correlation among features and with target.
- Located and resolved missing values.

After Third Pass — Where are we?

- Reviewed each feature — location, spread, shape, issues.
- Identified any correlation among features and with target.
- Located and resolved missing values.
- Generated list of possible feature engineering tasks.