SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

# DATA ENGINEERING AND ANALYTICS

Professor: JESÚS GARCÍA SAN LUIS
E-mail: jgarcias@faculty.ie.edu

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

# Data Preparation

# SESSION 2

# Data Science Projects

| Bussiness Understanding | Data Understanding | Data Preparation | Modeling | Evaluation | Deployment |
| --- | --- | --- | --- | --- | --- |
| • Business Objectives<br>• Technical constraints<br>• Project Planning | • Raw Data<br>• Graphics<br>• Quality Verification | • Cleaning: empty values, outlayers<br>• Feature selection | • Model selection<br>• Parameters selection | • Technical evaluation<br>• Business utility | • Results presentation<br>• Application Architecture |

- Modeling is only 20% of the total effort
- As a consequence, Data Science is not about learning algorithms. Is about dealing with data in a comprehensive and systematic way to select the best algorithms and apply them correctly
- You must use a systematic approach to develop data science projects

# Know Your Data

- How many rows

- Per column (one dimension)
  - Type (categorical / numeric)
  - How many different values
  - Simple statistics (mean, median, mode, stddev, histograms)
  - Missing values
  - Mismatched

- Plot data in time (if series data)

- How is data distributed: histograms and density plots
- Plot correlation matrix
- Scatterplot matrix

- Make “reasonable” assumptions from business knowledge and check them

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY



# Know Your Data Using Pyhton

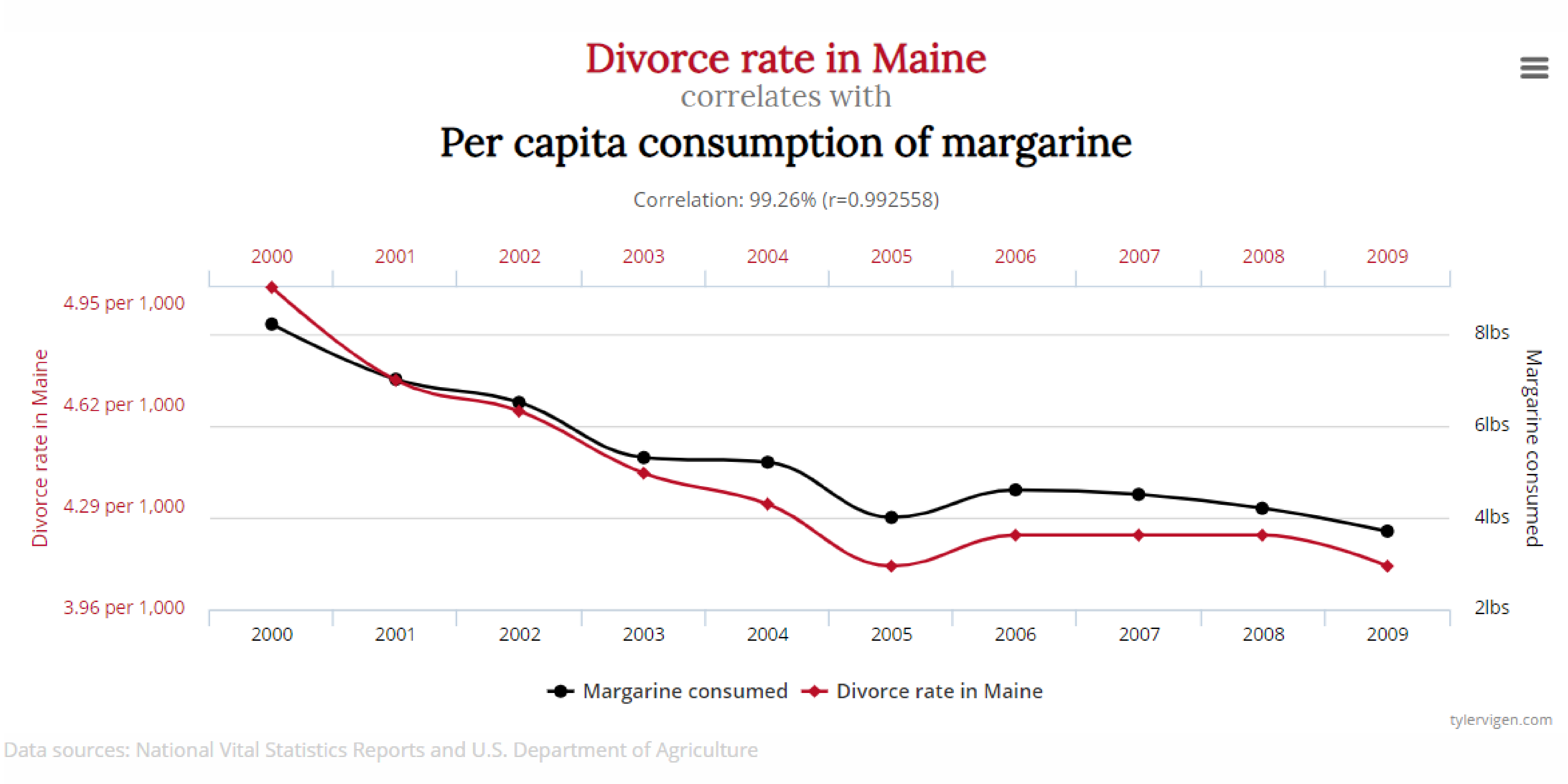
(See knowYourData.ipynb)

# Know Your Data

## A word about correlation

- Calculate correlation of this two variables:
  - `X = [0, 1, 3, 6, 9, 0, -1, -3, -6, -9]`
  - `Y = [0, 1, 3, 6, 9, 0,  1,  3,  6,  9]`
  - `np.corrcoef(X, Y)`

## Spurious Correlations



# Data Cleaning

- **Garbage in – Garbage out principle**

- **Generic pandas function for applying transformations**
  - `df.transform(f)` where f is a function defined in your code
  - `df.transform(lambda x: x + 1)`
  - `Usage for one column: df[‘column_name’] = df[‘column_name’].transform(…)`

- **Format / Spelling / Dates / Numbers**

- **Outliers**
  - One dimension – n dimensions
  - Statistical outlier detection: $\mu_i - \varepsilon\theta_i < f_i < \mu_i + \varepsilon\theta_i$
  - Clustering

- **Empty values**
  - Remove
  - Guess a value
    - Statistic value
    - Prediction model
    - Clustering
  - Let the model deal with empty values

# Feature Transformation

## Scaling

$$x_{scaled} = \frac{x}{max(|x|)}$$

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

$$x_{std} = \frac{x - \mu}{\sigma}$$

- **using pandas:**

```
df_scaled[column] = df[column]  / df[column].abs().max()
df_normalized[column] = (df[column] - df[column].min()) / (df[column].max() - df[column].min())
df_standardized[column] = (df_std[column] - df_std[column].mean()) / df_std[column].std()
```

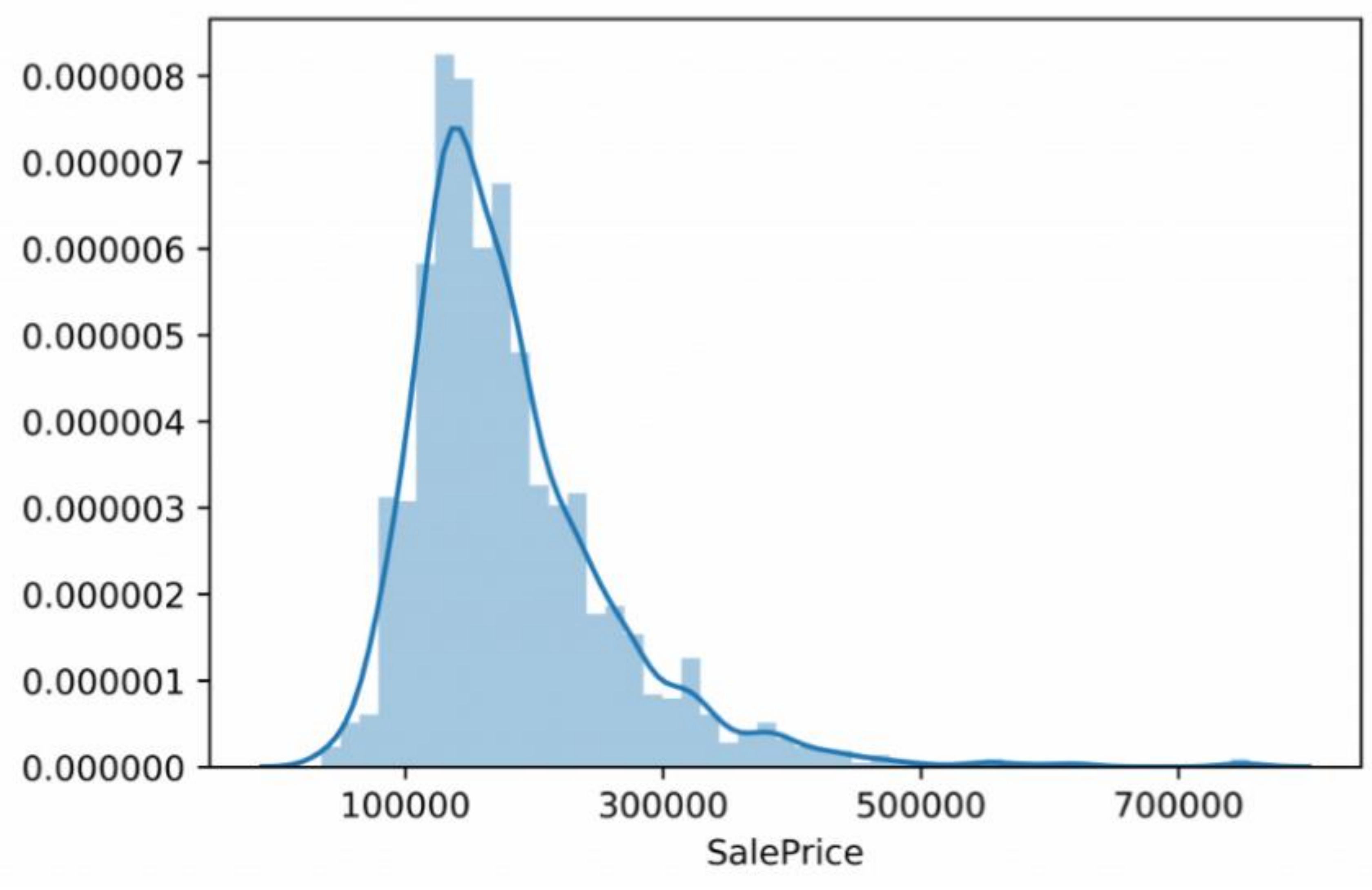
# Feature Transformation

## Centering data distributions

- **Skewness measure**

```
from scipy.stats import skew
print(skew(x))
```

- **Typical funcions for correcting skweness**
  Square root
  Reciprocal (1 / x)
  **Log(x)**



# Feature Transformation

**Category values encoding:** ordinal**,** frequency, binary

**One Hot Encoding:** Transform a categorical variable

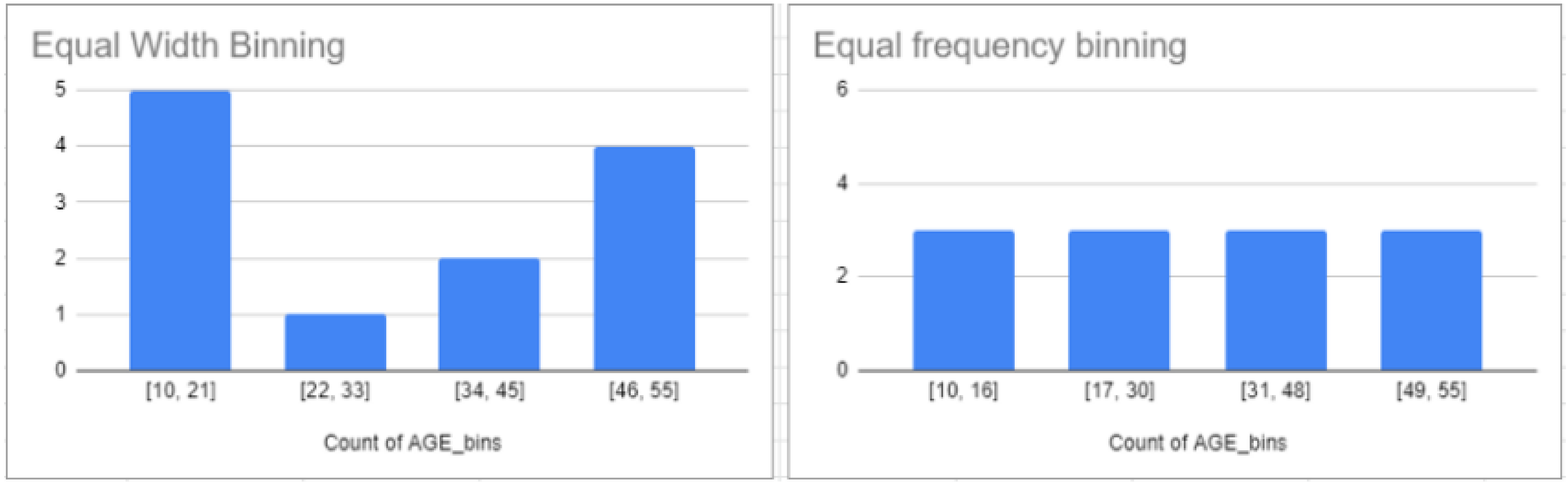
```
df = pd.get_dummies(df, prefix = [‘one_hot'], columns = [‘country'])
```

| | one_hot_australia | one_hot_china | one_hot_france | one_hot_japan | one_hot_spain |
|---|---|---|---|---|---|
| **0** | 0 | 0 | 0 | 0 | 1 |
| **1** | 0 | 0 | 1 | 0 | 0 |
| **2** | 1 | 0 | 0 | 0 | 0 |
| **3** | 0 | 0 | 0 | 1 | 0 |
| **4** | 0 | 1 | 0 | 0 | 0 |

# Feature Transformation

**Binning:** Transform a more or less continuos numerical variable into a categorical variable

- **Equal width**
```
data[“binnedAge"] = pd.cut(data[“Age"], bins=4)
```
- **Equal frequency**
```
data[“binnedAge"] = pd.qcut(data[“Age"], q=4)
```



# Feature Transformation

**Combine columns –** very dependent on domain knowledge

- Calculations with different columns (c1 x c2)
- Calculations with the same column (sqrt(c1))
- Difference between dates
- Group similar categorical values
- Temporal values

```
df.loc[i, ‘last_week_sales’] = df.loc[i-7, ‘sales']
```

# Unbalanced Datasets

- **Very often in classification datasets**
  - Is this transaction fraudulent?
  - Is this customer going to switch to a different company?
  - Has this patient this disease?

- **Why should we care?**

- **How to solve it**
  - Undersampling
  - Oversampling

  - Generate synthetic samples – SMOTE (Synthetic Minority Oversample)
    - Imbalanced-learn python library

```
oversample = SMOTE()
X, y = oversample.fit_resample(X, y)
```

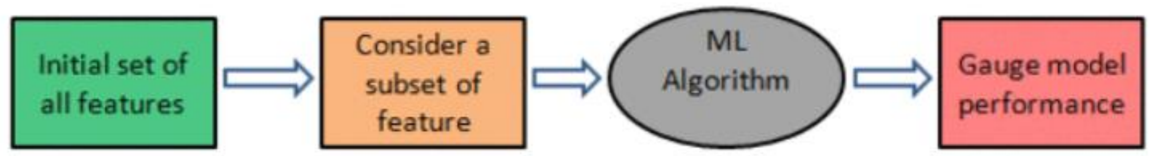
# Feature Selection

- **The principle: *“Brevity is the soul of wit” (William Shakespeare)***

- **Feature Selection**
  - Which inputs are relevant?
  - Why do we need to select inputs at all?

- **Some techniques for feature selection/elimination**
  - Irrelevant inputs (i.e., ID, Passport Number…)
  - Low variance inputs
  - High correlation among inputs
  - Univariate feature selection: correlation with output

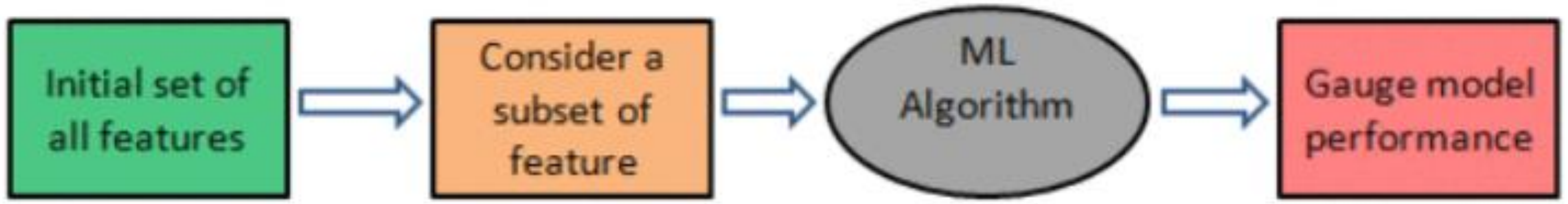
  - Wrapper: Step Forwards, Step Backwards, Exhaustive




  - Embedded Methods: Ridge and Lasso regressors. Trees

# Feature Selection

- Wrapper Methods: Step Forwards, Step Backwards, Exhaustive




- Embedded Methods:
  - Lasso / Ridge

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}|\beta_j|$$

  - Trees

```
forest = RandomForestClassifier(random_state=0)
forest.fit(X_train, y_train)
importances = forest.feature_importances_
```

# Feature Selection

**Principal Component Analysis**: change your point of view

- Or a little more formally: choose a new orthogonal base where the variance along each direction is maximized
- Be aware: you have to rotate the data forth and back to get interpretable results. Many times the new base has not real meaning