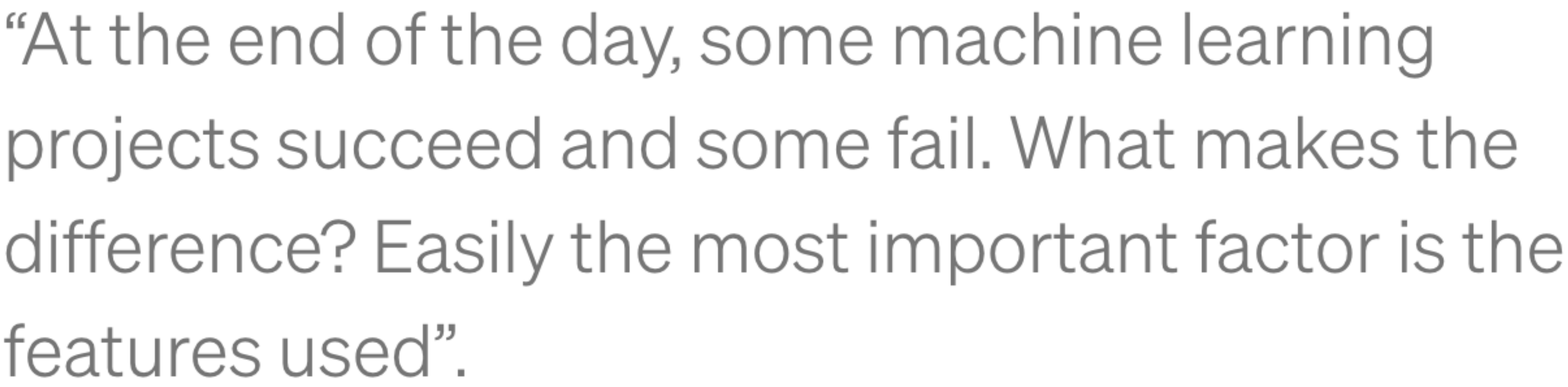
Feature Engineering

- What is feature engineering?
- Why do we need to engineer features?
- What FE methods do we need?
  - Feature Encoding
  - Missing Data Imputation
  - Numeric Data Transformation
- Take-home messages

Feature Engineering

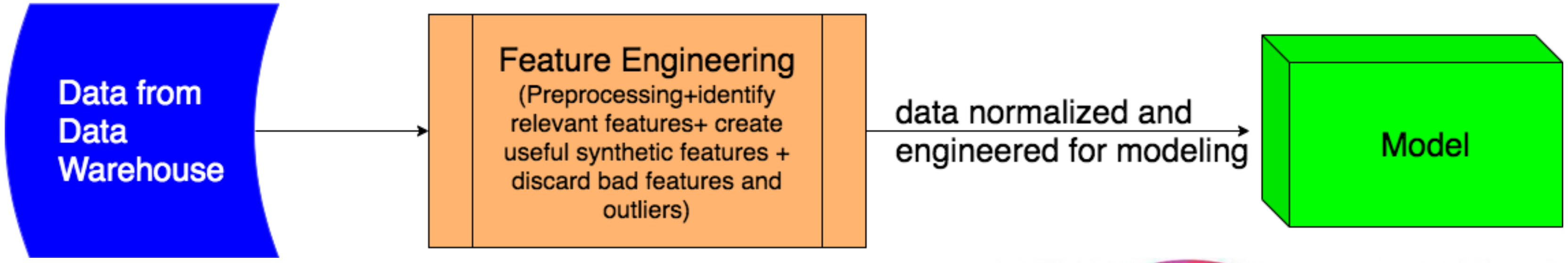


"At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used".

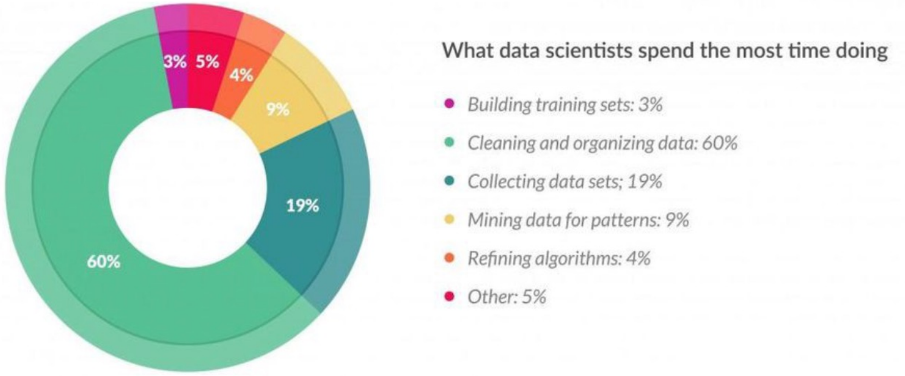
Pedro Domingos

What is feature engineering (FE) about?

- What is a feature and why we need the engineering of it?
- Machine learning and data mining algorithms use some input data (comprising features) to create outputs.
- Goals of FE:
  - Preparing the proper input dataset, compatible with the machine learning algorithm requirements.
  - Improving the performance of machine learning models.

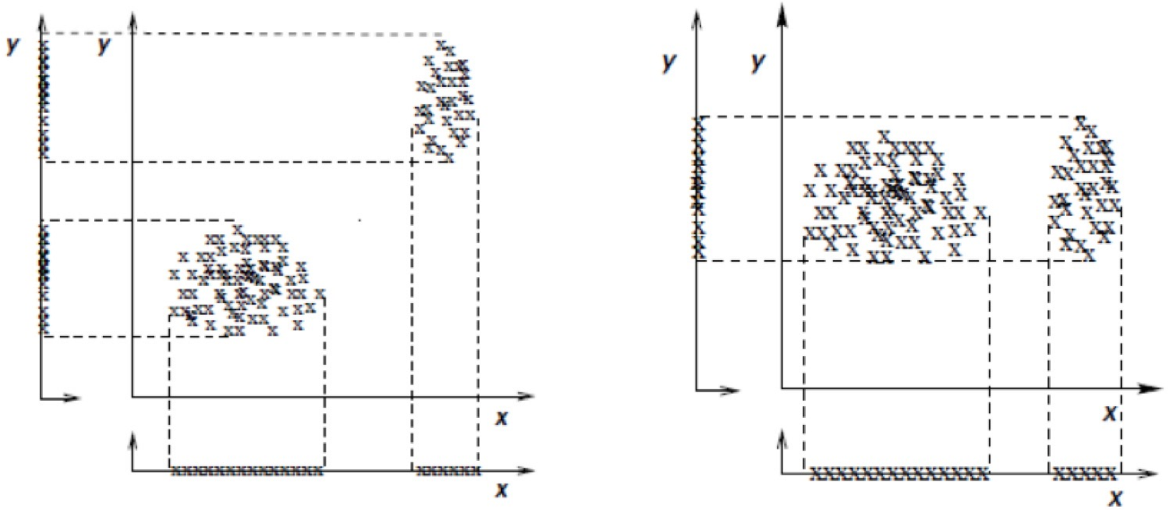

Data from Data Warehouse
Feature Engineering
(Preprocessing+identify relevant features+ create useful synthetic features + discard bad features and outliers)
data normalized and engineered for modeling
Model

- According to a survey in Forbes, data scientists spend **80%** of their time on **data preparation:**



Feature Engineering-Science or Art?

- Feature Engineering is the process of selecting and extracting useful, predictive signals from data.
- Scientifically,

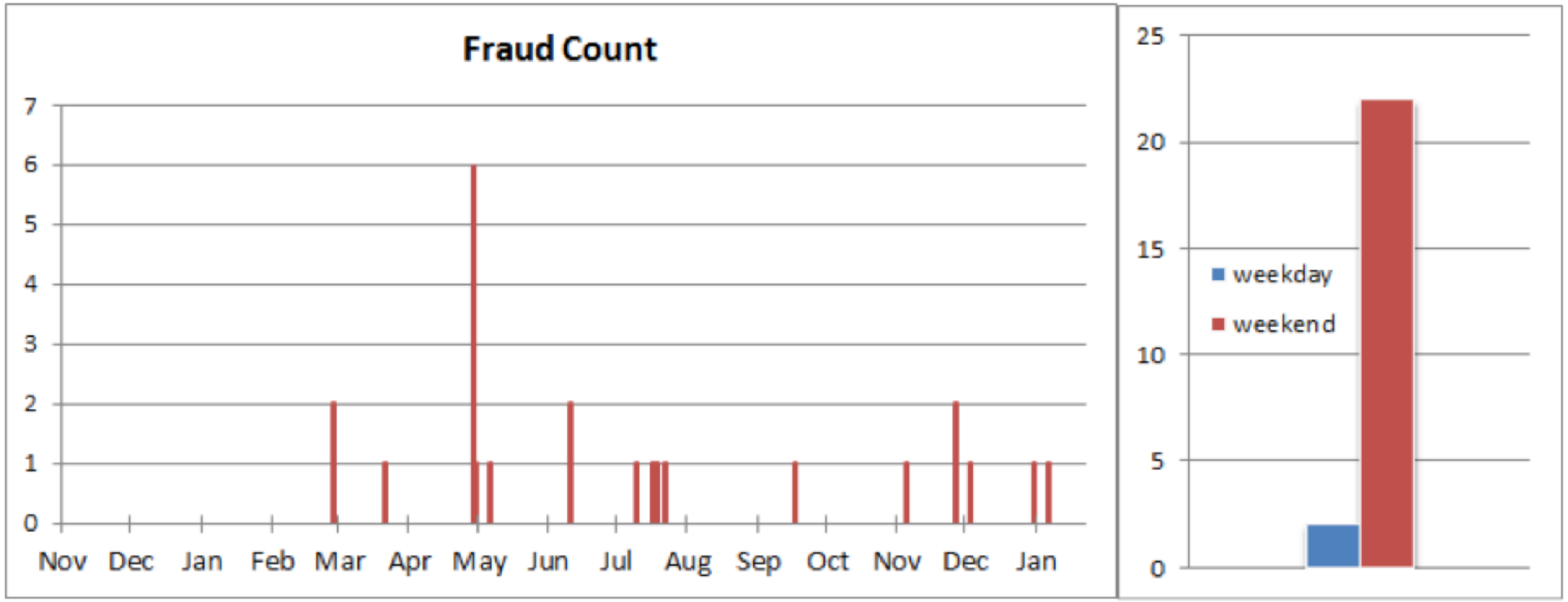


*Example of redundant feature (left: feature **y** provides the same information as **x**) and irrelevant feature (right: feature **y** does not discriminate between the clusters defined by **x**). http://www.jmlr.org/papers/volume5/dy04a/dy04a.pdf*

Click here to see more!

Feature Engineering-Science or Art? (cont.)

- How can you conceive the use of day-of-the-week feature for the following cybersecurity example (monitoring of fraudulent activity)?



See more details from here!

Timeline of fraudulent activity (left) and its breakdown by day of the week (right)

Why do we need to engineer features?

1. Some machine learning libraries do not support missing values or strings as inputs, for example Scikit-learn.
2. Some machine learning models are sensitive to the magnitude of the features.
3. Some algorithms are sensitive to outliers.
4. Some variables provide almost no information in their raw format, for example dates
5. Often variable pre-processing allows us to capture more information, which can boost algorithm performance!
6. Frequently variable combinations are more predictive than variables in isolation, for example the sum or the mean of a group of variables.
7. Some variables contain information about transactions, providing time-stamped data, and we may want to aggregate them into a static view.

More reasons behind…

- The most important factor of ML/DM project is the features used.
  - With many independent features that each correlate well with the predicted class, learning is easy.
  - On the other hand, if the class is a very complex function of the features, you may not be able to learn it.
- ML/DM is often the quickest part, but that’s because we’ve already mastered it pretty well!
  - Feature engineering is more difficult because it’s domain-specific, while learners (ML/DM models) can be largely general-purpose.
- Today it is often done by automatically generating large numbers of candidate features and selecting the best by (say) their information gain with respect to the predicted class. Bear in mind that features that look irrelevant in isolation may be relevant in combination.
- On the other hand, running a learner with a very large number of features to find out which ones are useful in combination may be too time-consuming, or cause overfitting.

Experience it in your competition and report it!

Yet some final comments about FE

More data beats a cleverer algorithm

- ML researchers are mainly concerned with creation of better learning algorithm, but pragmatically the quickest path to success is often to just get more data.
- This does bring up another problem, however: scalability.
  - In most of computer science, the 2 main limited resources are time and memory. In ML and DM, there is a 3rd one: training data.

    ➔ Data is the King!
- **Try simpler algorithms/models first** as more sophisticated ones are usually harder to use (more tuning before getting good results).

What exactly do I need to learn?

1. Missing data imputation
2. Categorical variable encoding (e.g. one-hot encoding: link1, link2)
3. Numerical variable transformation
4. Discretization
5. Engineering of datetime variables
6. Engineering of coordinates — GIS data
7. Feature extraction from text
8. Feature extraction from images
9. Feature extraction from time series
10. New feature creation by combining existing variables

Only some of them are required by the house price prediction competition. Please visit Kaggle's discussions and Medium's resources for learning FE.

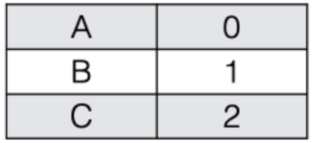
Feature Encoding Methods

- Turn categorical features into numeric features to provide more fine-grained information
- Help explicitly capture non-linear relationships and interactions between the values of features
- Most of machine learning tools only accept numbers as their input, e.g., XGBoost, gbm, glmnet, libsvm, liblinear, etc.
- Labeled encoding, one-hot encoding, frequency encoding, etc.

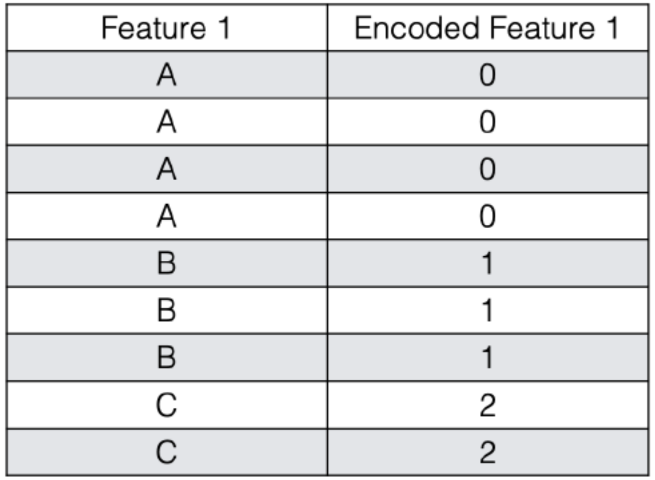
Feature Encoding Methods (cont.)

Labeled Encoding

- Interpret the categories as ordered integers (mostly wrong)
- Python scikit-learn: LabelEncoder
- Ok for tree-based methods



A	0
B	1
C	2



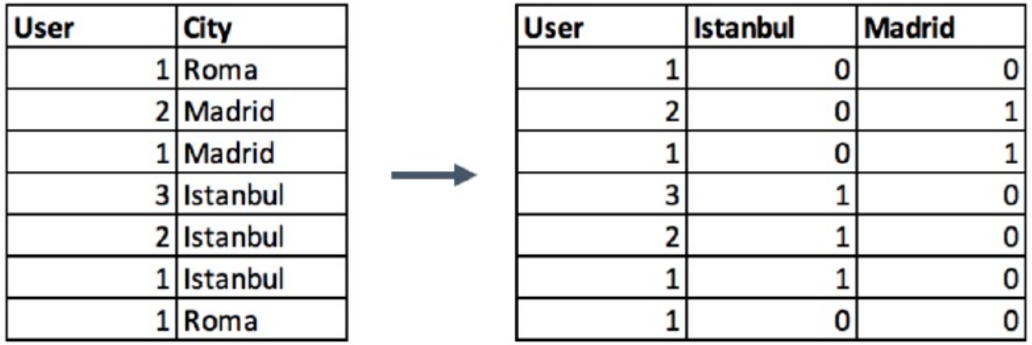
Feature 1	Encoded Feature 1
A	0
A	0
A	0
A	0
B	1
B	1
B	1
C	2
C	2

Feature Encoding Methods (cont.)

One-hot Encoding

- Transform categories into individual binary (0 or 1) features
- Python scikit-learn: DictVectorizer, OneHotEncoder

This method spreads the values in a column to multiple flag columns and assigns 0 or 1 to them.

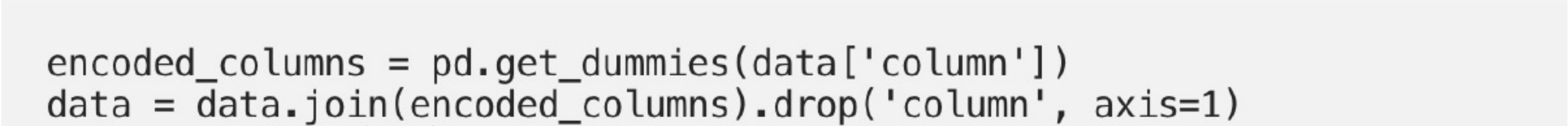


User	City
1	Roma
2	Madrid
1	Madrid
3	Istanbul
2	Istanbul
1	Istanbul
1	Roma

→

User	Istanbul	Madrid
1	0	0
2	0	1
1	0	1
3	1	0
2	1	0
1	1	0
1	0	0

One hot encoding example on City column

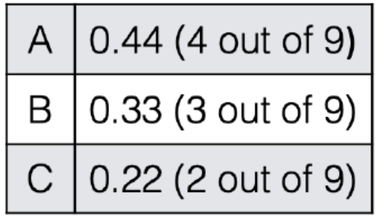


```
encoded_columns = pd.get_dummies(data['column'])
data = data.join(encoded_columns).drop('column', axis=1)
```

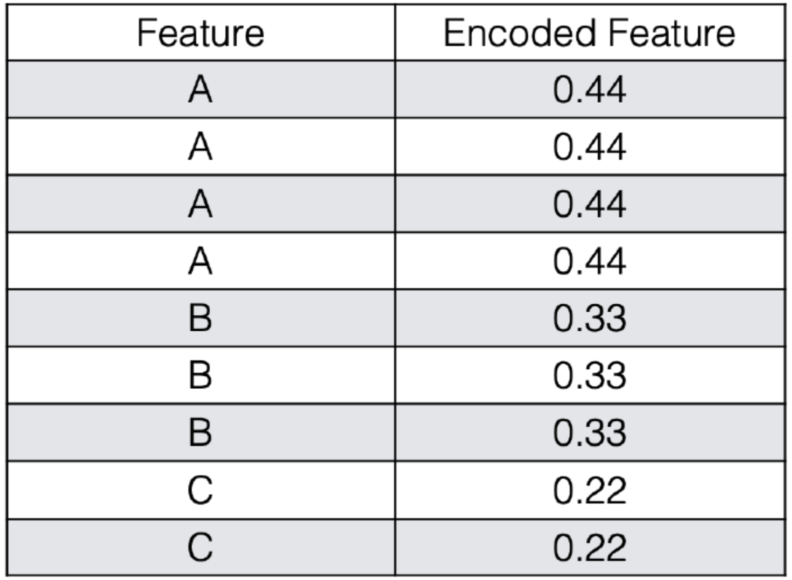
Feature Encoding Methods (cont.)

Frequency encoding

- Encoding of categorical levels of feature to values between 0 and 1 based on their relative frequency



A	0.44 (4 out of 9)
B	0.33 (3 out of 9)
C	0.22 (2 out of 9)



Feature	Encoded Feature
A	0.44
A	0.44
A	0.44
A	0.44
B	0.33
B	0.33
B	0.33
C	0.22
C	0.22

Missing Data Imputation

- Common problem in preparing the data: Missing Values
- Data is not always available
    - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
    - equipment malfunction
    - inconsistent with other recorded data and thus deleted
    - data not entered due to misunderstanding
    - certain data may not be considered important at the time of entry
    - not register history or changes of the data
- Missing data may need to be inferred!! But how?

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing; but is not good for attribute values (assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably)
- Fill in the missing value manually: tedious + infeasible?
- Use a global constant to fill in the missing value: e.g., “unknown”, a new class?!
- Use the attribute mean to fill in the missing value
- Use the attribute mean for all samples belonging to the same class to fill in the missing value: smarter
- Use the most probable value to fill in the missing value: inference-based such as Bayesian formula or decision tree and associative-based

Guessing the missing data (Aggarwal et al., KDD2006)

Name	Title	Gender	M.Status	Education	Salary Level
Amy	Assistant	F	Unmarried	HD	SL-3
Bobby	Assistant	M	Married	HD	SL-3
Catherine	Assistant	F	Married	University	SL-3
Don	Manager	F	Unmarried	University	SL-5
Elaine	Manager	F	Married	University	SL-5
Franky	Manager	M	Married	University	SL-5
Grace	Manager	F	Married	M.B.A.	SL-7
Helen	Manager	F	Married	Ph.D	SL-5
Ivan	Accountant	F	Unmarried	M.B.A.	SL-5
Jenny	Accountant	M	Married	University	SL-4

So, the following is resulted.

Name	Title	Gender	M.Status	Education	Salary Level
Amy	Assistant	F	Unmarried	HD	SL-3
Bobby	Assistant	M	Married	HD	SL-3
Catherine	Assistant	F	Married	University	🟩
Don	Manager	F	Unmarried	University	SL-5
Elaine	Manager	F	Married	University	SL-5
Franky	Manager	M	Married	🟩	SL-5
Grace	Manager	F	🟩	M.B.A.	SL-7
Helen	Manager	F	Married	Ph.D	SL-5
Ivan	Accountant	🟩	Unmarried	M.B.A.	SL-5
Jenny	Accountant	🟩	Married	University	SL-4

- Catherine wants to hide her salary level
- Franky wants to hide his education background
- Grace wants to hide her marriage status
- Ivan wants to hide his gender
- Jenny wants to hide her gender as well

Guessing the missing data (Aggarwal et al., KDD2006)

By apply association analysis to the dataset with missing data, we can obtain

- ❑ R1: Assistant → SL-3 (support:2, confidence=100%)
- ❑ R2: Manager ∧ SL-5 → University (support:2, confidence=66.7%)
- ❑ R3: Manager ∧ Female → Married (support:2, confidence=66.7%)

So, we can guess the missing value as follows.

- ❑ Catherine's salary level is SL-3
- ❑ Franky's education is university
- ❑ Grace's marriage status is married

(Numeric) Data Transformation

Different Forms of Transformation

- Smoothing: remove noise from data
- Aggregation: summarization, data cube construction
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
  - min-max normalization
  - z-score normalization
  - normalization by decimal scaling
- Attribute/feature construction
  - New attributes constructed from the given ones

Most commonly used transformation: Normalization

- min-max normalization

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

- z-score normalization

$$v' = \frac{v - mean_A}{std\_dev_A}$$

- normalization by decimal scaling

$$v' = \frac{v}{10^j}$$

where j is the smallest integer such that max(|v'|)≤1

Take-home Messages

- Feature engineering is related to data preprocessing, which will be further discussed in data warehousing.
- FE covers many other aspects/methods not elaborated here, e.g., feature extraction for text, image, time series, etc. -> application dependent!
- As pointed out by Prof. Pedro Domingos, FE is critical to the success of data mining projects.