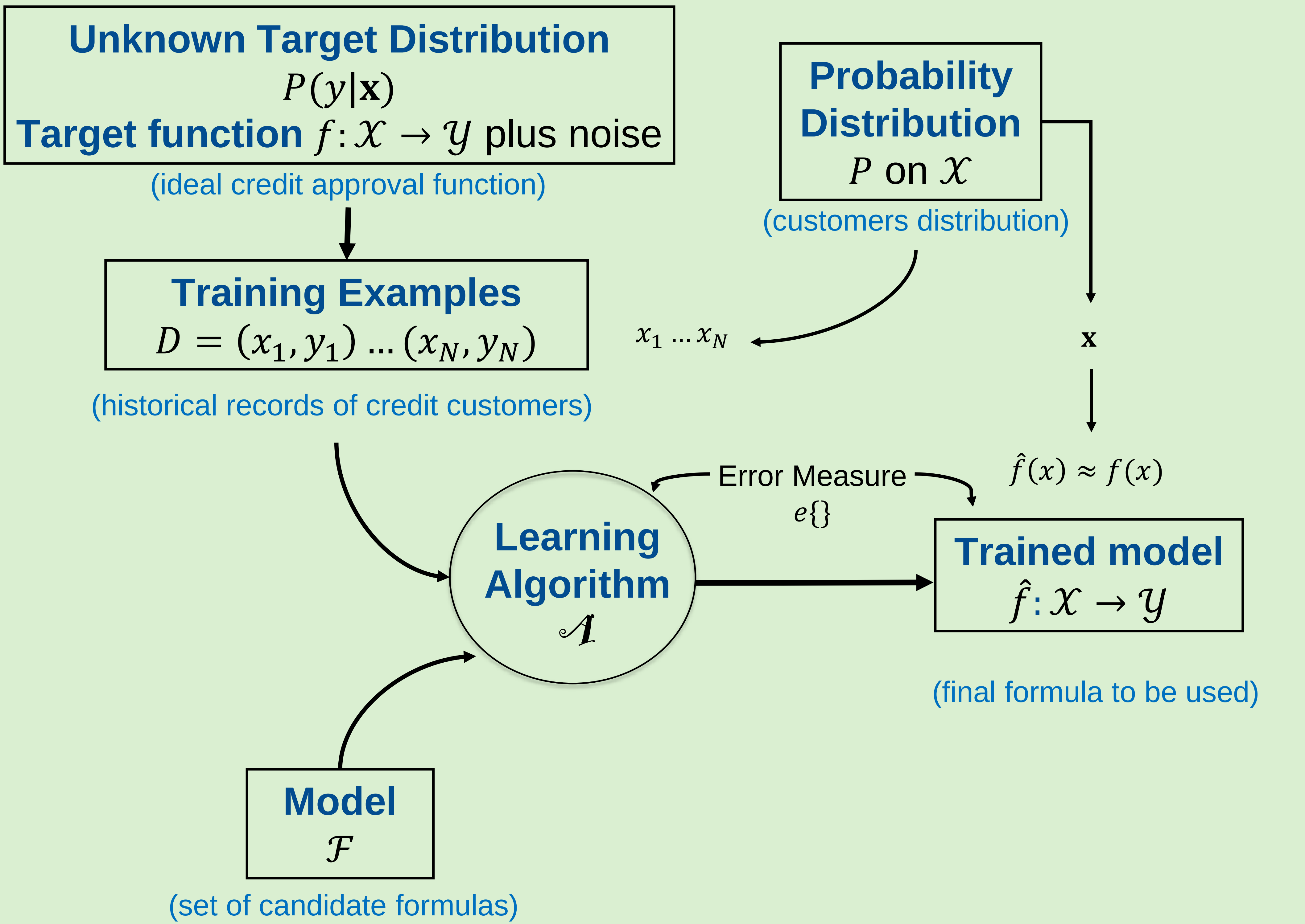
Introduction to Machine Learning

Dor Bank

Lecture:
ML project & preprocessing

The learning diagram – ML course



Today - CRISP DM

- CRoss Industry Standard Process for Data Mining
- Several other diagrams exists, but this is the most common
- Real data is not simply sampled from $P(y|\mathbf{x})$!
- Business context, missing values, etc…
- The key is “to tell the data story”

Business understanding

- What problem are we trying to solve?
- Decide on clear measures to assess the success of the project
- For our course project, this does not take much place, as it is already defined for you

Data Understanding / Exploration

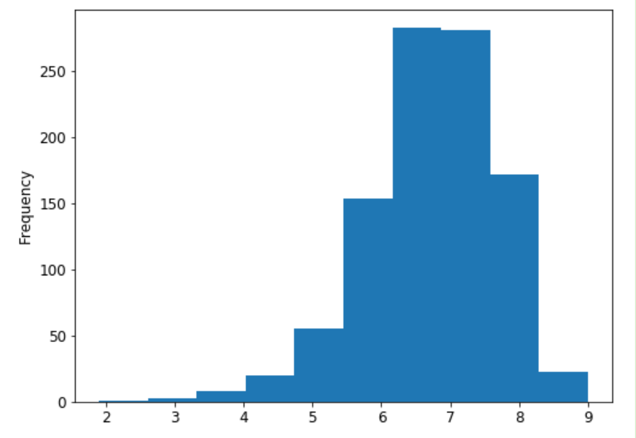
- Usually known as EDA – Exploratory data analysis
- Business perspective – what is the meaning of each feature? How do we expect each feature to correlate with the others? With the labels? etc.
- Statistics & visualization:
  - How does the data distributes?
  - What can we learn from the data?
  - BIG room from visualizations!
  - Plot for purpose, not for plotting ☺

Data Understanding / Exploration

- Distribution examples:

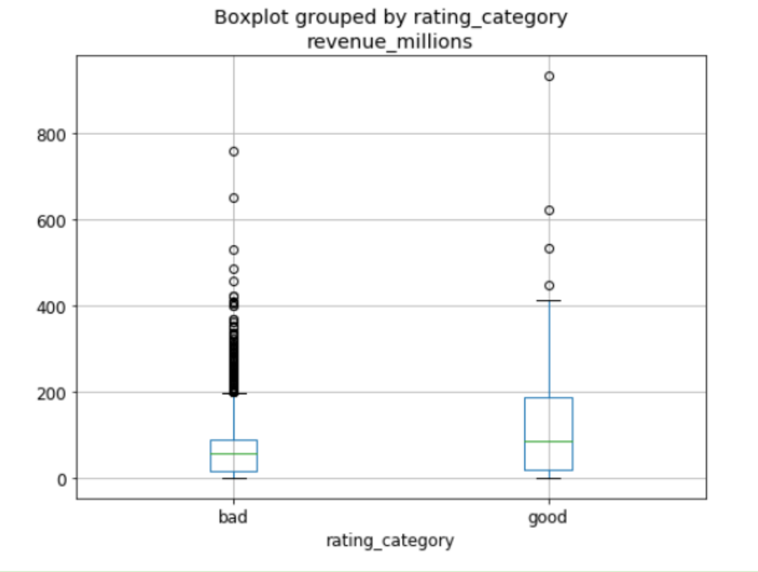
Histogram

```
df['rating'].plot.hist()
plt.show()
```



Boxplots

```
df.boxplot(column='revenue_millions',
           by='rating_category');
plt.show()
```



Statistics

```
df.describe()
```

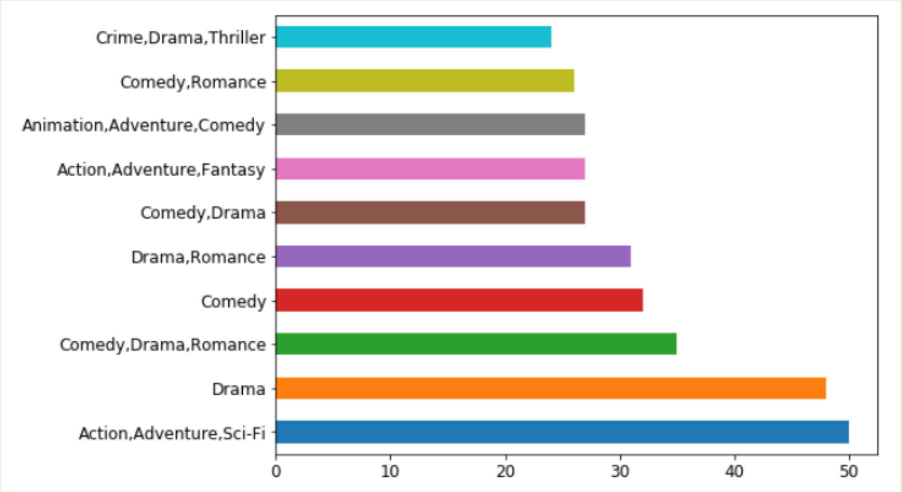
	year	runtime	rating
count	1000.000000	1000.000000	1000.000000
mean	2012.783000	113.172000	6.723200
std	3.205962	18.810908	0.945429
min	2006.000000	66.000000	1.900000
25%	2010.000000	100.000000	6.200000
50%	2014.000000	111.000000	6.800000
75%	2016.000000	123.000000	7.400000
max	2016.000000	191.000000	9.000000

Data Understanding / Exploration

- More examples:

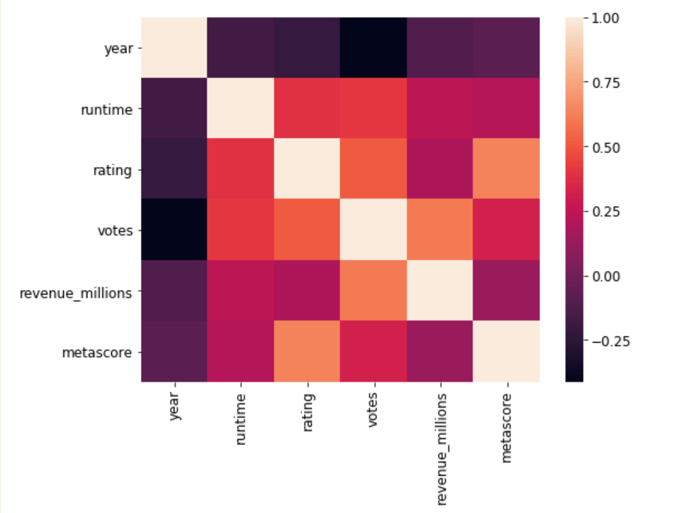
Categorical Features Distributions

```
df['genre'].value_counts().head(10).plot.barh()
plt.show()
```



Features Correlation

```
sns.heatmap(df.corr(),
         xticklabels=df.corr().columns,
         yticklabels=df.corr().columns)
plt.show()
```



Missing Values

```
df.isnull().sum()
```

```
title          0
genre          0
description    0
director       0
actors         0
year           0
```

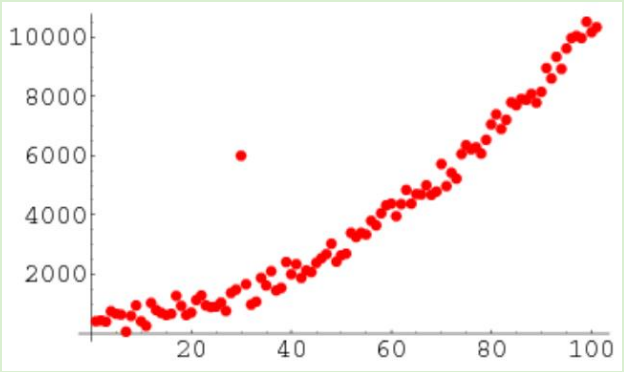
Data Preparation / Preprocessing

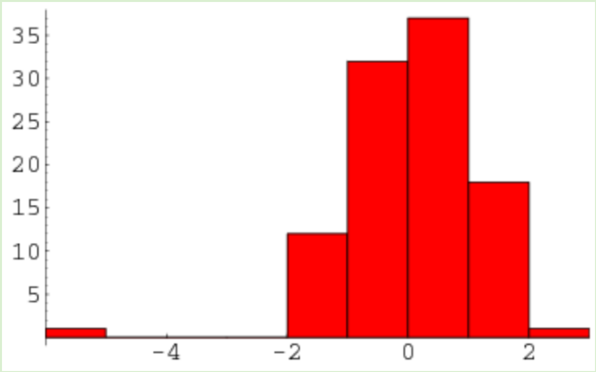
- After we get a good understanding, we start to process the data a get it ready for the ML model
- It basically includes:
  - Outlier removal
  - Filling missing values
  - Dimensionality reduction
  - Data transformation / normalization
  - Feature engineering
  - etc.

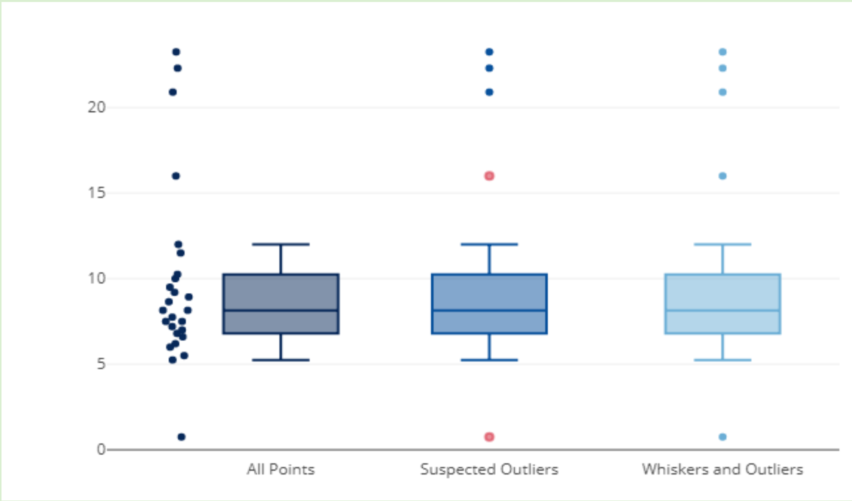
Preprocessing - Outlier removal

- These are samples that do not represent the true distribution of the data, and we do not want our model to learn from them
- Various methods for doing so
- Example – using boxplots (assuming the data distributes normally)
- TIP: Do not forget to plot the data!!
- Visualization usually helps

```
from scipy import stats
df  = df[(np.abs(stats.zscore(df)) < 3).all(axis=1)]
```







if $\boldsymbol{x_{ik}} < \boldsymbol{Q_1} - \mathbf{1.5} \cdot \boldsymbol{IQR}$ or $\boldsymbol{x_{ik}} > \boldsymbol{Q_3} + \mathbf{1.5} \cdot \boldsymbol{IQR}$

$\rightarrow \boldsymbol{Outlier}$

Preprocessing – filling missing values

- How does a missing value look like?
- A sample/feature with many missing values – remove it
- Fill missing values
  - By average \ median (numerical)
  - By most frequent \ new category (categorical)
  - Constant / zero
  - Serialized data (i.e with dates) – use the previous and the next
  - KNN imputation – use the nearest neighbors data

```python
revenue = df["revenue_millions"]
revenue.isnull().sum()
```

```
128
```

```python
revenue_mean = revenue.mean()
revenue.fillna(revenue_mean, inplace=True)
revenue.isnull().sum()
```

```
0
```

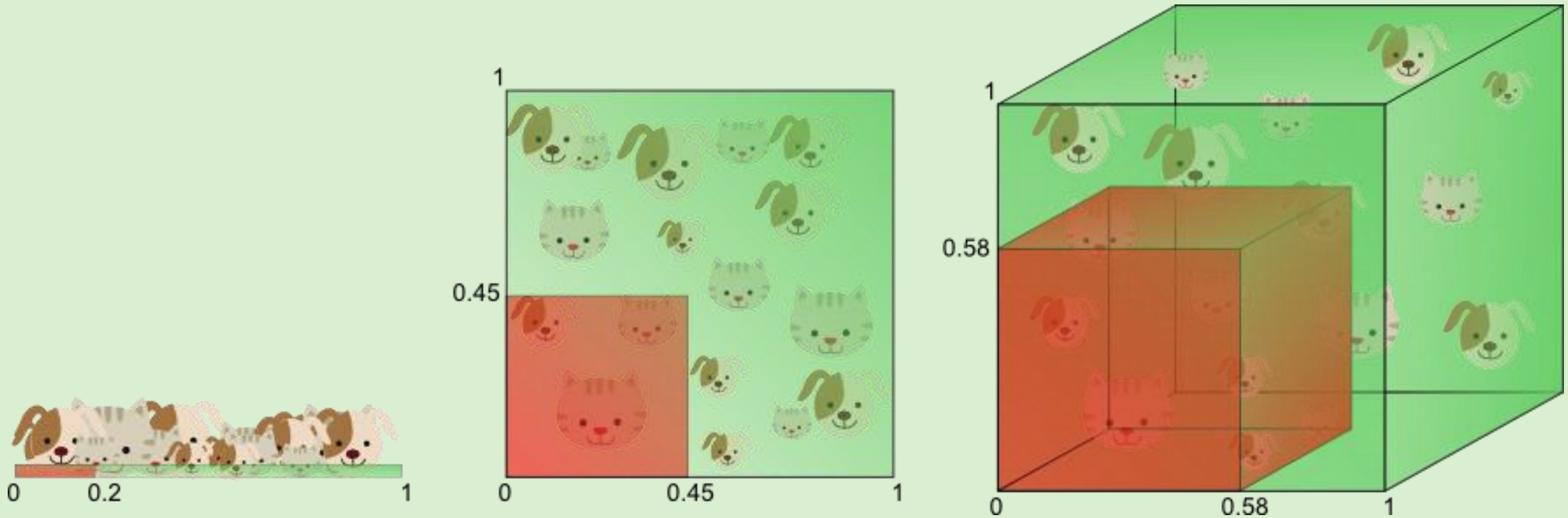
	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

df.fillna(0)

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	0.0
1	9	0.0	9.0	0	7.0
2	19	17.0	0.0	9	0.0

Preprocessing – dimensionality reduction

- Why reduce the dimensionality?
  - Increases model variance
  - Exposure to more noise than signal
  - Curse of dimensionality – the space is sparser



Feature selection types

- **Filter method:** Ranks features or feature subsets independently of the classifier
  - Low computational power
  - Independent of model type

Now ☺

- **Wrapper method:** Uses a predictive model (machine learning) to score feature subsets
  - Requires training a model for each feature set
  - Commonly used AFTER filter methods

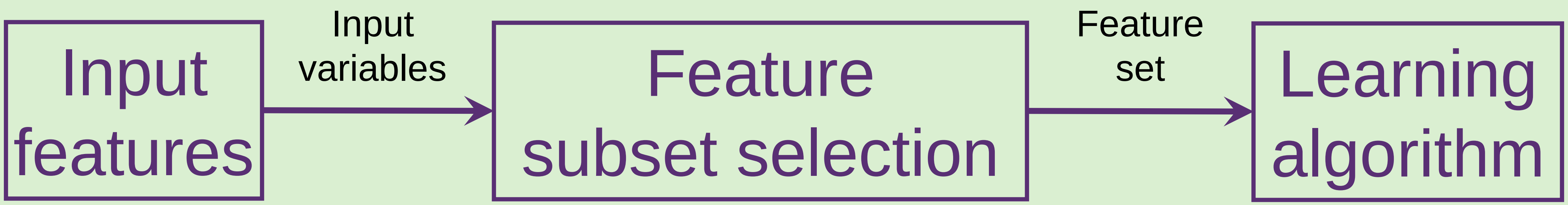
Lecture 4

- **Embedded method:** Performs variable selection (implicitly) in the course of model training (e.g. decision tree\Lasso)

Meet along the course

Feature selection – Filter methods

- Select subsets of variables as a pre-processing step, by ranking according to some scoring metric, **independently of the learning model**



- Relatively fast & not tuned by a given learner
- Very commonly used

Feature selection – Filter methods

- Examples:
  - Label association:
    - Example: Choose the top 10 features correlated with the label
  - Low variation (sparse) features:
    - Remove features with little variation in their value
  - Correlated features (redundancy):
    - Keep only one out of two highly correlated features

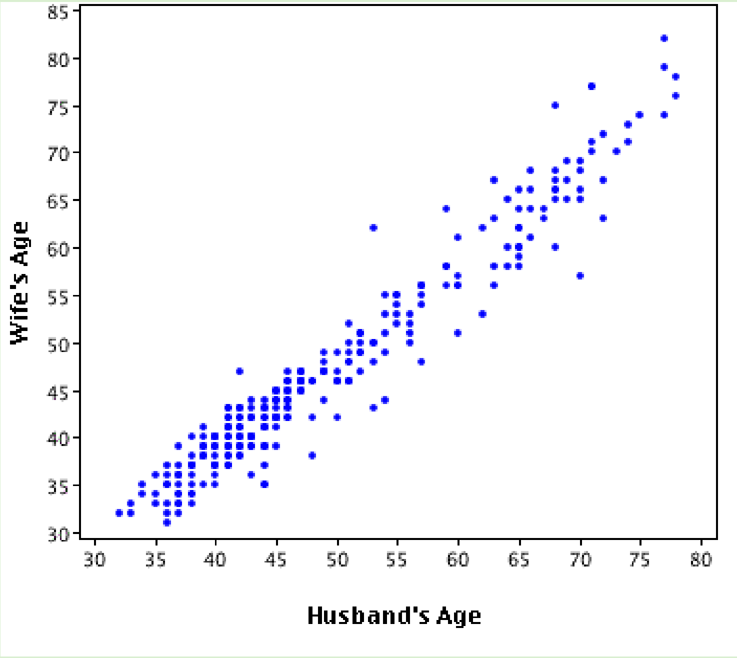
Feature selection – Filter methods: scoring functions

- Pearson correlation
  - Measures the linear relationship between two features [continuous not categorical]
  - Definition : $\rho_{X,Y} = \frac{Cov(X,Y)}{SD(X)*SD(Y)} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$
  - Sample Correlation definition:

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} * \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

  - Range: [-1,1] (what does the -1,0,1 values mean?)
  - No correlation is not necessarily independent. The opposite is true however.
  - If r(X,Y) = 0.8, what does it means?

In python: scipy.stats.pearsonr



Feature selection – Filter methods: scoring functions

- Mutual information
  - Measures the amount of uncertainty in X which is removed by knowing Y
  - Discrete random variables X and Y:

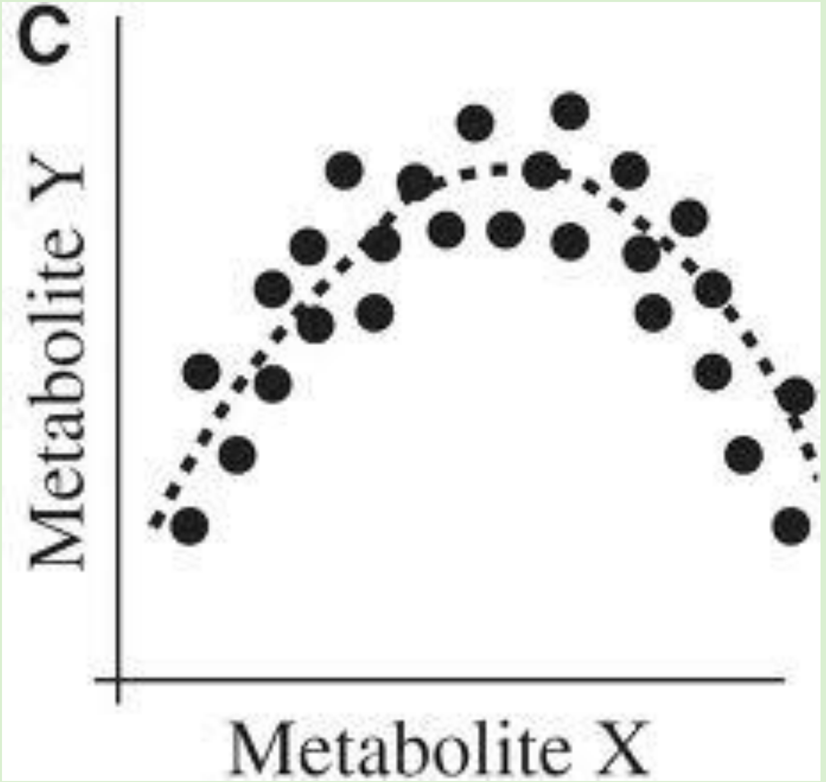
$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right)$$

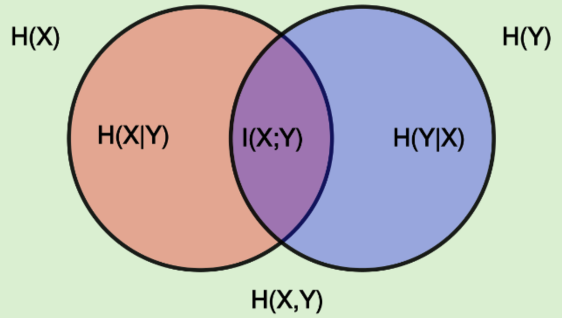
  - Continuous random variables X and Y

$$I(X,Y) = \int_X \int_Y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) dydx$$

  - Non negative (equal 0 if X, Y are independent)
  - Isn't restricted to linear dependency
  - Works for both discrete and continuous variables

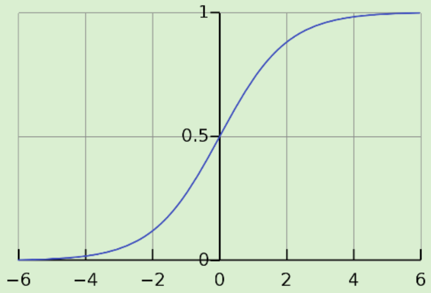
In python: Sklearn.metrics.mutual_info_score





Preprocessing – Data transformation

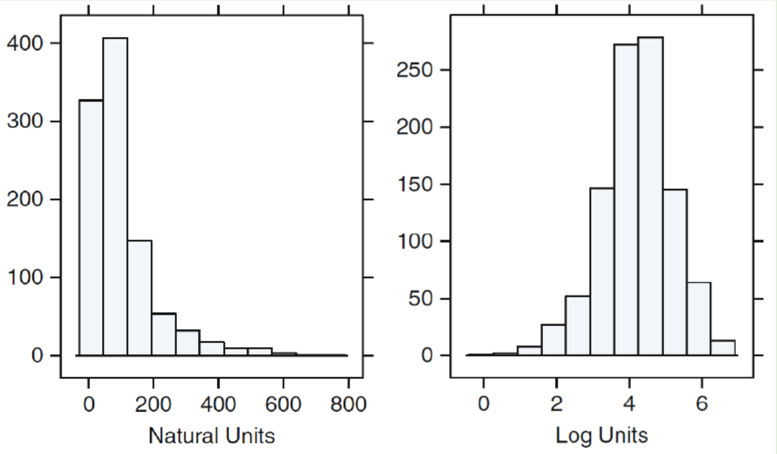
- Normalization : 10 years is not like 10,000$....
  - **0-1**: make all features between 0 and 1 by reducing the minimal value and dividing by the max
    - Bounded, but sensitive to outliers
  - **Sigmoid**: $x' = \frac{1}{1+e^{-\alpha x}}$



    - Bounded, less sensitive, almost linear at the center, but squeezes the edges
  - **Standardize**: make all features to have 0 mean and 1 variance be reducing the mean and dividing by the variance.
    - Highly intuitive, but not bounded

Preprocessing – Data transformation

- **<u>Box & Cox</u>**: used to reduce the skewness



$$x^* = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(x) & \text{if } \lambda = 0 \end{cases}$$

$(\lambda = 0.5)$	$(\lambda = -1)$	$(\lambda = 2)$
Square root transformation	*Inverse transformation*	*Square transformation*

- **<u>OneHotEncoding / Dummy variables</u>** – turning categorical features to numerical

Product Usage Category	*Original Variable Code*	*<u>Dummy Variable Code</u>* D1	D2	D3
Nonusers...............	1	1	0	0
Light Users...........	2	0	1	0
Medium Users.......	3	0	0	1
Heavy Users..........	4	0	0	0

$$\hat{Y}_i = a + b_1D_1 + b_2D_2 + b_3D_3$$

- **<u>Discretization</u>** – turning numerical features to categorical
  - $x' = 0 \; if \; x > 10.12.2005 \; else \; 1$

Preprocessing – Data transformation

- In practice – we usually use sklearn transformers
- Instead of ‘fit’ and ‘predict’, we have ‘fit’ and ‘transform’
  - The ‘fit’ learns needed parameters (like mean/variance for StandardScaler, or categories for OneHotEncoder)
- **DO NOT USE ‘fit’ ON THE TEST DATA!**
- To be extra careful, do not even fit on the validation in model selection

Standardization
(or Z-score normalization)

$$\hat{x}_{ik} = \frac{x_{ik} - \bar{x}_k}{\bar{\sigma}_k}$$

```
from sklearn.preprocessing import StandardScaler

 # We initialize our scaler
standard_scaler = StandardScaler()

# We fit our scaler
standard_scaler.fit(X)

# We transform our X using the scaler we have just fit.
scaled_X = standard_scaler.transform(X)
```

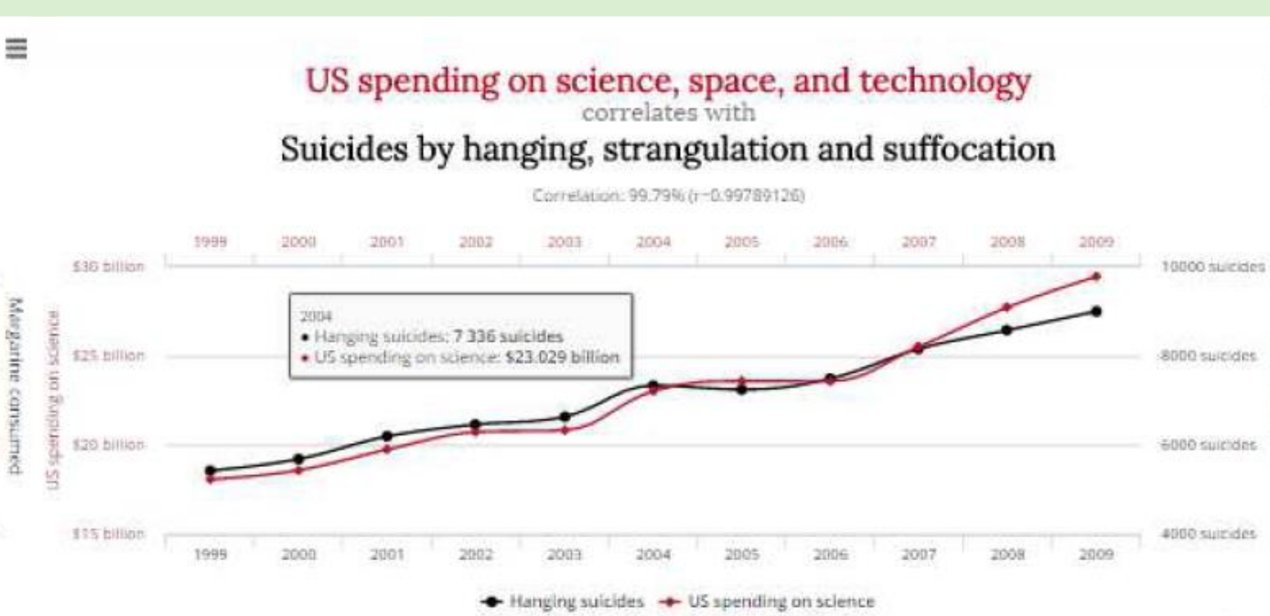
MinMax Scaling

$$\hat{x}_{ik} = \frac{x_{ik} - \min(x_k)}{\max(x_k) - \min(x_k)}$$

```
from sklearn.preprocessing import MinMaxScaler

# We initialize our scaler
min_max_scaler = MinMaxScaler()

# We fit our scaler
min_max_scaler_scaler.fit(X)

# We transform our X using the scaler we have just fit.
scaled_X = min_max_scaler.transform(X)
```

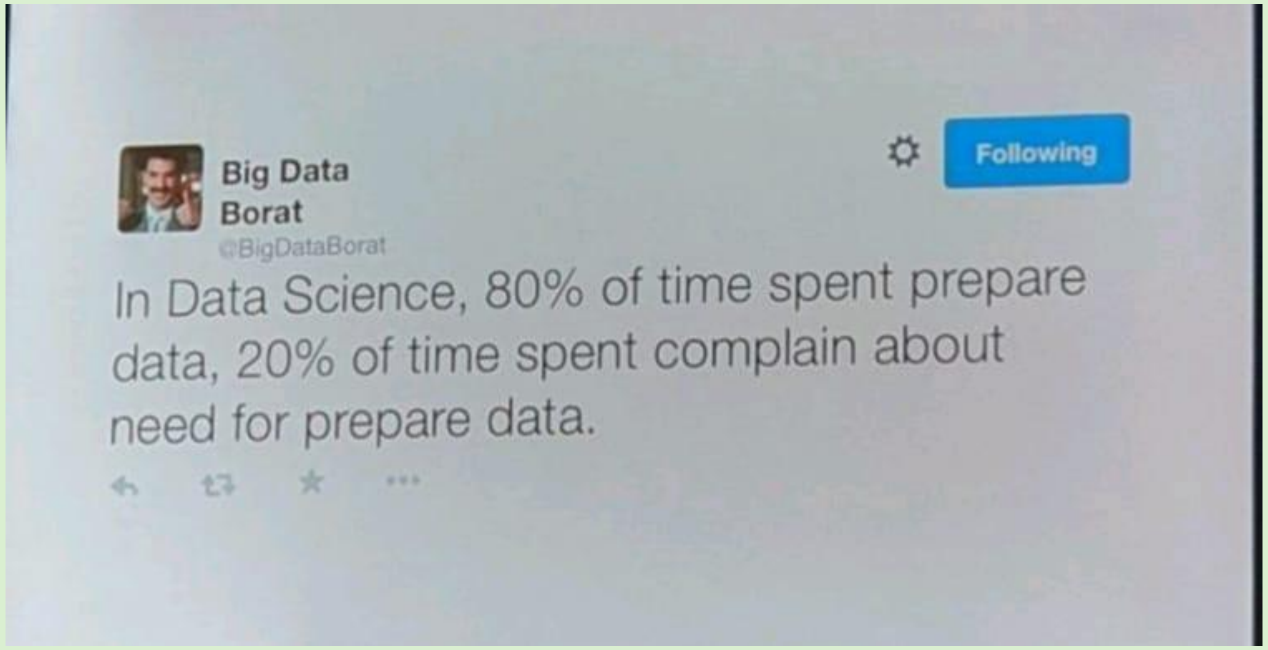
Preprocessing – Feature engineering

- Create new features for the existing ones (instead / on top)
- PCA is an example for both dimensionality reduction & feature engineering
- Others maybe used with business understanding / domain knowledge
- **Examples**
  - Dates -> day of week
  - Weight & Height -> BMI
  - Grade1, grade2, grade3 -> grade average
  - Etc.
- Word of caution – correlation does not imply causation



Data Preparation / Preprocessing

- We have covered some examples
- Feel free to use other methods which make sense
- A lot of room for creativity
- Key to success!



Modeling

- Try out different models
- Did our preprocessing match the model?
  - Example: using mutual information and using linear model
- Make sure to “exploit” the full capacity of each model
  - Hyper parameter tuning
  - Regularization
  - etc.

Evaluation

- Check the performance of the model
- Validation, Cross Validation
- Notice the difference between the loss function and the business metric
  - For example, minimizing the cross entropy VS maximizing AUC

Deployment

- After we are sure about our results we “go to production”
- In practice, Not that simple!
- In our project, it simply means to submit predictions for the test set

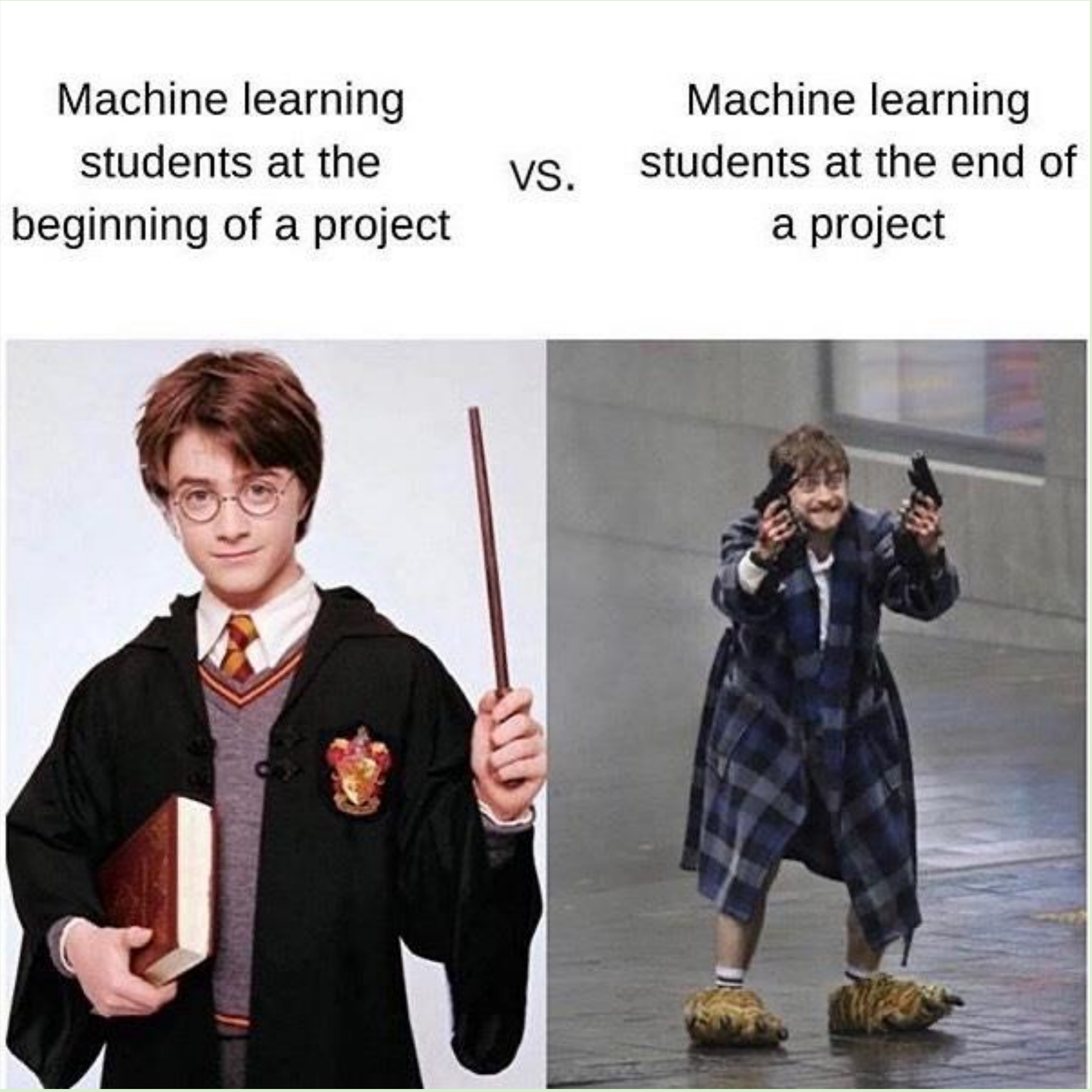
Important notes

- The CRISP-DM is cyclic, with the ability to go back!
- A DS is an iterative process
- Tip: get as fast and naively through the first iteration
  - Get a first benchmark for evaluation
  - Get past technical difficulties
  - Get initial insight on the data
  - This holds for our project and in general!

Important notes

- For our project, a great project is one where the notebook is almost not needed
- The report (and the results) tell complete story

Good Luck!


Machine learning students at the beginning of a project
vs.
Machine learning students at the end of a project