

Today

Recap: Top-down vs bottom-up

Predicting things

You have done it before!

From regression to classification

Supervised Machine Learning for Text Classification

((Traditional)) non-SML approaches

Diving into SML

An implementation


Classifiers

Vectorizers

Summing up


Revisiting the difference between the dictionary approach and

Recap

	Methodological approach		
	<i>Counting and Dictionary</i>	<i>Supervised Machine Learning</i>	<i>Unsupervised Machine Learning</i>
Typical research interests and content features	visibility analysis sentiment analysis subjectivity analysis	frames topics gender bias	frames topics
Common statistical procedures	string comparisons counting	support vector machines naive Bayes	principal component analysis cluster analysis latent dirichlet allocation semantic network analysis
			

Boumans and Trilling, 2016

The same logic applies to non-textual data!

	Methodological approach		
	<i>Counting and Dictionary</i>	<i>Supervised Machine Learning</i>	<i>Unsupervised Machine Learning</i>
Typical research interests and content features	visibility analysis sentiment analysis subjectivity analysis	frames topics gender bias	frames topics
Common statistical procedures	string comparisons counting	support vector machines naive Bayes	principal component analysis cluster analysis latent dirichlet allocation semantic network analysis
			

Boumans and Trilling, 2016

The same logic applies to non-textual data!

Some terminology

Supervised machine learning

You have a dataset with both predictor and outcome (independent and dependent variables; features and labels) — a *labeled* dataset. Think of regression: You measured x_1 , x_2 , x_3 and you want to predict y , which you also measured

Unsupervised machine learning

You have no labels.

Some terminology

Supervised machine learning

You have a dataset with both predictor and outcome (independent and dependent variables; features and labels) — a *labeled* dataset. Think of regression: You measured x_1 , x_2 , x_3 and you want to predict y , which you also measured

Unsupervised machine learning

You have no labels.

Some terminology

Supervised machine learning

You have a dataset with both predictor and outcome (independent and dependent variables; features and labels) — a *labeled* dataset. Think of regression: You measured x_1 , x_2 , x_3 and you want to predict y , which you also measured

Unsupervised machine learning

You have no labels. (You did not measure y)

Again, you already know some techniques to find out how x_1 , x_2, \dots, x_i co-occur from other courses:

- Principal Component Analysis (PCA) and Singular Value Decomposition (SVD)
- Cluster analysis
- Topic modelling (Latent Dirichlet Allocation)

Some terminology

Supervised machine learning

You have a dataset with both predictor and outcome (independent and dependent variables; features and labels) — a *labeled* dataset. Think of regression: You measured x_1 , x_2 , x_3 and you want to predict y , which you also measured

Unsupervised machine learning

You have no labels. (You did not measure y)

Again, you already know some techniques to find out how x_1 , x_2, \dots, x_i co-occur from other courses:

- Principal Component Analysis (PCA) and Singular Value Decomposition (SVD)
- Cluster analysis
- Topic modelling (Latent Dirichlet Allocation)

Some terminology

Supervised machine learning

You have a dataset with both predictor and outcome (independent and dependent variables; features and labels) — a *labeled* dataset. Think of regression: You measured x_1 , x_2 , x_3 and you want to predict y , which you also measured

Unsupervised machine learning

You have no labels. (You did not measure y)

Again, you already know some techniques to find out how x_1 , x_2, \dots, x_i co-occur from other courses:

- Principal Component Analysis (PCA) and Singular Value Decomposition (SVD)
- Cluster analysis
- Topic modelling (Latent Dirichlet Allocation)

Predicting things

Predicting things

You have done it before!

You have done it before!

Regression

- $$\hat{y}_{woman40} = -.8 + .4 \times 0 + .08 \times 40 = 2.4$$

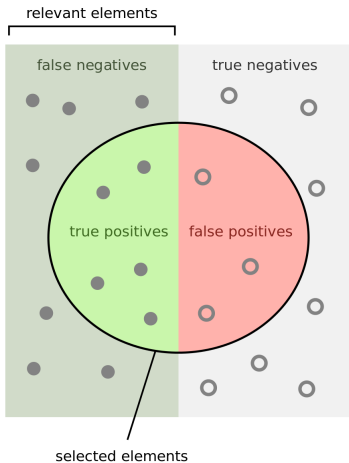
This is
Supervised Machine Learning!

Predicting things

From regression to classification

In the machine learning world, predicting some continuous value is referred to as a **regression** task. If we want to predict a binary or categorical variable, we call it a **classification** task.

(quite confusingly, even if we use a logistic regression for the latter)



How many selected
items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant
items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Some measures

- Accuracy
- Recall
- Precision
- $F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$
- AUC (Area under curve)
[0, 1], 0.5 = random
guessing

Different classification algorithms

- It is an empirical question which one works best
- We typically try several ones and select the best
- (remember: we have a test dataset that we did *not* use to train the model, so that we can assess how well it predicts the test labels based on the test features)
- To avoid *p*-hacking-like scenario's (which we call "overfitting"), there are techniques available (cross-validation, later in this course)

(to make it easier, imagine a binary classification ("positive"/"negative"), but it doesn't really matter whether there are two or more labels)

- (to make it easier, imagine a binary classification ("positive"/"negative"), but it doesn't really matter whether there are two or more labels)

Different classification algorithms

- It is an empirical question which one works best
- We typically try several ones and select the best
- (remember: we have a test dataset that we did *not* use to train the model, so that we can assess how well it predicts the test labels based on the test features)
- To avoid *p*-hacking-like scenario's (which we call "overfitting"), there are techniques available (cross-validation, later in this course)

(to make it easier, imagine a binary classification ("positive"/"negative"), but it doesn't really matter whether there are two or more labels)

- It is an empirical question which one works best
- We typically try several ones and select the best
- (remember: we have a test dataset that we did *not* use to train the model, so that we can assess how well it predicts the test labels based on the test features)
- To avoid p -hacking-like scenario's (which we call "overfitting"), there are techniques available (cross-validation, later in this course)

(to make it easier, imagine a binary classification ("positive"/"negative"), but it doesn't really matter whether there are two or more labels)

Naïve Bayes

Bayes' theorem

$$P(A | B) = \frac{P(B | A) \times P(A)}{P(B)}$$

A = Text is about sports

B = Text contains 'very', 'close', 'game'. Furthermore, we simply multiply the probabilities for the features:

$$P(B) = P(\text{very close game}) = P(\text{very}) \times P(\text{close}) \times P(\text{game})$$

We can fill in all values by counting how many articles are about sports, and how often the words occur in these texts. (Fully

elaborated example on

<https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/>)

Naïve Bayes

Bayes' theorem

$$P(A | B) = \frac{P(B | A) \times P(A)}{P(B)}$$

A = Text is about sports

B = Text contains 'very', 'close', 'game'. Furthermore, we simply multiply the probabilities for the features:

$$P(B) = P(\text{very close game}) = P(\text{very}) \times P(\text{close}) \times P(\text{game})$$

We can fill in all values by counting how many articles are about sports, and how often the words occur in these texts. (Fully

elaborated example on

<https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/>)

Bayes' theorem

$$P(A | B) = \frac{P(B | A) \times P(A)}{P(B)}$$

B = Text contains 'very', 'close', 'game'. Furthermore, we simply multiply the probabilities for the features:

$$P(B) = P(\text{very close game}) = P(\text{very}) \times P(\text{close}) \times P(\text{game})$$

We can fill in all values by counting how many articles are about sports, and how often the words occur in these texts. (Fully

elaborated example on

<https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/>

1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 26

- It's "naïve" because the features are treated as completely independent (\neq "controlling" in regression analysis)
- It's fast and easy
- It's a good *baseline* for binary classification problems

Naïve Bayes

- It's "naïve" because the features are treated as completely independent (\neq "controlling" in regression analysis)
- It's fast and easy
- It's a good *baseline* for binary classification problems

Naïve Bayes

$$P(\text{label} \mid \text{features}) = \frac{P(x_1 \mid \text{label}) \cdot P(x_2 \mid \text{label}) \cdot P(x_3 \mid \text{label}) \cdot P(\text{label})}{P(x_1) \cdot P(x_2) \cdot P(x_3)}$$

- Formulas always look intimidating, but we only need to fill in how many documents containing feature x_n have the label, how often the label occurs, and how often each feature occurs
- Also for computers, this is *really easy and fast*
- Weird assumption: features are independent
- Often used as a baseline

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)}}$$

Just like in OLS regression, we have an intercept and regression coefficients. We use a threshold (default: 0.5) and above, we assign the positive label ('good movie'), below, the negative label ('bad movie').

Logistic Regression

- The features are *not* independent.
- Computationally more expensive than Naïve Bayes
- We can get probabilities instead of just a label
- That allows us to say how sure we are for a specific case
- ...or to change the threshold to change our precision/recall-tradeoff

100

- The features are *not* independent.
- Computationally more expensive than Naïve Bayes
- We can get probabilities instead of just a label
- That allows us to say how sure we are for a specific case
- ... or to change the threshold to change our precision/recall-tradeoff

Logistic Regression

- The features are *not* independent.
- Computationally more expensive than Naïve Bayes
- We can get probabilities instead of just a label
 - That allows us to say how sure we are for a specific case
 - ...or to change the threshold to change our precision/recall-tradeoff

Logistic Regression

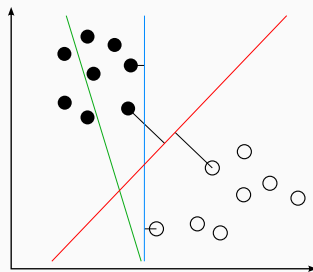
- The features are *not* independent.
- Computationally more expensive than Naïve Bayes
- We can get probabilities instead of just a label
- That allows us to say how sure we are for a specific case
- ... or to change the threshold to change our precision/recall-tradeoff

Logistic Regression

- The features are *not* independent.
- Computationally more expensive than Naïve Bayes
- We can get probabilities instead of just a label
- That allows us to say how sure we are for a specific case
- ...or to change the threshold to change our precision/recall-tradeoff

Support Vector Machines

- Idea: Find a hyperplane that best separates your cases
- Can be linear, but does not have to be (depends on the so-called kernel you choose)
- Very popular



https:

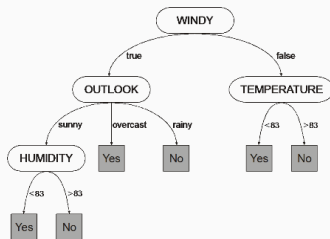
[//upload.wikimedia.org/wikipedia/commons/b/b5/Svm_separating_hyperplanes %28SVG%29.svg](https://upload.wikimedia.org/wikipedia/commons/b/b5/Svm_separating_hyperplanes_%28SVG%29.svg)

(Further reading: <https://www.oxfordjournals.org/doi/10.1093/oxfordjournals/medntr.a011002>)

[//monkeylearn.com/blog/introduction-to-support-vector-machines-svm/](https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/)

Decision Trees and Random Forests

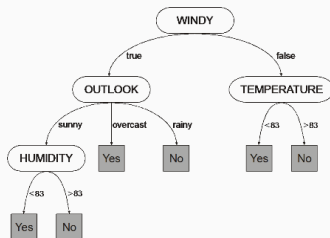
- Model problem as a series of decisions (e.g., if cloudy then ... if temperature > 30 degrees then ...)
- Order and cutoff-points are determined by an algorithm
- Big advantage: Model non-linear relationships
- And: They are easy to interpret (!) ("white box")



https://upload.wikimedia.org/wikipedia/en/4/4f/GEP_decision_tree_with_numeric_and_nominal_attributes.png

Decision Trees and Random Forests

- Model problem as a series of decisions (e.g., if cloudy then ... if temperature > 30 degrees then ...)
- Order and cutoff-points are determined by an algorithm
- Big advantage: Model non-linear relationships
- And: They are easy to interpret (!) (“white box”)



https://upload.wikimedia.org/wikipedia/en/4/4f/GEP_decision_tree_with_numeric_and_nominal_attributes.png

19

Supervised Machine Learning for Text Classification

Supervised Machine Learning for Text Classification

(Traditional)) non-SML approaches

topic e.g., [sports|economy|politics|entertainment|other]

frames e.g., [economic|human|moral|conflict], or

non-exclusive: economic = $[0|1]$, human = $[0|1]$, ...



What would be the strengths and weaknesses of different approaches for each of these tasks?



Imagine using a dictionary-based (list of keywords, list of regular expressions, or similar) approach to these tasks. How does the design (length, inclusiveness, etc.) of this list influence precision and recall?

Dictionary-based approaches for text classification

good for

- distinct, manifest things
(names of organizations,
pronouns, swearwords (?),
...)
- little room for interpretation/misunderstandings etc.
- “must-be-explainable-to-a-five-year-old”

bad for

- latent constructs and concepts
- implicit things

Hence, *not* state-of-the-art for

- topics
- frames
- sentiment

Dictionary-based approaches for text classification

good for

- distinct, manifest things
(names of organizations,
pronouns, swearwords (?),
...)
- little room for interpreta-
tion/misunderstandings etc.
- “must-be-explainable-to-a-
five-year-old”

bad for

- latent constructs and
concepts
- implicit things

Hence, *not* state-of-the-art for

- topics
- frames
- sentiment

good for

- distinct, manifest things
(names of organizations,
pronouns, swearwords (?),
...)
- little room for interpreta-
tion/misunderstandings etc.
- “must-be-explainable-to-a-
five-year-old”

bad for

- latent constructs and concepts
- implicit things

Hence, *not* state-of-the-art for

- topics
- frames
- sentiment

From dictionary approaches to SML

- Early days of sentiment analysis: list of positive words, list of negative words, count what occurs most
- You can even *buy* lists of words that are meant to measure constructs like “positive emotions” or even “analytic” or “authentic” language use from a psychologist (LIWC, Pennebaker et al., 2007)

From dictionary approaches to SML

- Early days of sentiment analysis: list of positive words, list of negative words, count what occurs most
- You can even *buy* lists of words that are meant to measure constructs like “positive emotions” or even “analytic” or “authentic” language use from a psychologist (LIWC, Pennebaker et al., 2007)



What do you think? Can this even work

Bag-of-words dictionary approaches to sentiment analysis

con

- simplistic assumptions
- e.g., intensifiers cannot be interpreted (“really” in “really good” or “really bad”)
- or, even more important, negations.

Improving the BOW approach

Example: Sentistrength (Thelwall et al., 2012)

- $-5 \dots -1$ and $+1 \dots +5$ instead of positive/negative
- spelling correction
- “booster word list” for strengthening/weakening the effect of the following word
- interpreting repeated letters (“baaaaaad”), CAPITALS and !!!
- idioms
- negation

VADER by Hutto and Gilbert, 2014 works in a similar way. Even though this is much less naïve than LIWC, for instance, the problem remains: Can we construct a dictionary that, *irrespective of the context*, gives us a meaningful estimate of sentiment?

Improving the BOW approach

Example: Sentistrength (Thelwall et al., 2012)

- $-5 \dots -1$ and $+1 \dots +5$ instead of positive/negative
- spelling correction
- “booster word list” for strengthening/weakening the effect of the following word
- interpreting repeated letters (“baaaaaad”), CAPITALS and !!!
- idioms
- negation

VADER by Hutto and Gilbert, 2014 works in a similar way. Even though this is much less naïve than LIWC, for instance, the problem remains: Can we construct a dictionary that, *irrespective of the context*, gives us a meaningful estimate of sentiment?

Such an *off-the-shelf* dictionary does not
(and probably cannot) exist.

28

Boukes et al., 2020: Sentiment analysis of economic news

Table A1. Correlations between sentiment scores using different methods for headlines (above) and full texts (below).

	Headline							
	Manual coding	Recession	D & B	LIWC	SentiStrength	Pattern	Polyglot	DANEW
Manual coding	1.00 ***							
Recession	-	-						
Damstra and Boukes (2018)	0.16 ***	-	1.00 ***					
LIWC	0.30 ***	-	0.16 ***	1.00 ***				
SentiStrength	0.24 ***	-	0.08 **	0.26 ***	1.00 ***			
Pattern	0.22 ***	-	0.00	0.30 ***	0.22 ***	1.00 ***		
Polyglot	0.30 ***	-	0.19 ***	0.32 ***	0.37 ***	0.26 ***	1.00 ***	
DANEW	0.24 ***	-	0.04	0.43 ***	0.33 ***	0.23 ***	0.32 ***	1.00 ***
	Full text							
	Manual coding	Recession	D & B	LIWC	SentiStrength	Pattern	Polyglot	DANEW
Manual coding	1.00 ***							
Recession	-0.06 *	1.00 ***						
Damstra and Boukes (2018)	0.27 ***	-0.16 ***	1.00 ***					
LIWC	0.39 ***	0.02	0.27 ***	1.00 ***				
SentiStrength	0.17 ***	-0.01	0.10 ***	0.18 ***	1.00 ***			
Pattern	0.13 ***	-0.02	0.04	0.28 ***	0.12 ***	1.00 ***		
Polyglot	0.26 ***	0.05	0.17 ***	0.41 ***	0.21 ***	0.30 ***	1.00 ***	
DANEW	0.15 ***	0.06 *	0.05	0.36 ***	0.18 ***	0.29 ***	0.37 ***	1.00 ***

The word "recession" did not occur in headlines of our sample, as such, no correlation coefficient is available for the recession classifier; *** $p < .001$, ** $p < .010$, * $p < .05$.

Boukes et al., 2020: Sentiment analysis of economic news

- Dictionaries have low agreement with each other, and also with human coders
- Even their own dictionary didn't agree
- **This is not because these dictionaries are particularly bad!**. Main point: For such a complex and context-dependent task, a dictionary is just not the right tool.

van Atteveldt et al., 2021: Extending Boukes et al., 2020 with SML

“manual coding (using undergraduate students) yields the best results

[...] A good second place is taken by crowd coding [...]

[...] machine learning performs worse than both students' manual coding and crowd coding. Reaching $\alpha = 0.50$ for deep learning (CNN) and slightly worse for classical machine learning (SVM; $\alpha = 0.41$, NB; $\alpha = 0.40$), machine learning still performs significantly better than chance. However, since these results are lower than generally accepted levels of inter-coder reliability [...]

Finally, [...] dictionaries [...] perform worse than the machine learning results and much worse than manual annotation [...]

[and] approximate chance agreement”

SML is no panacea, but the most promising approach to analyzing large quantities of texts. Don't believe off-the-shelf packages that claim to do the work for you. (For small datasets, just do it by hand.)

Supervised Machine Learning for Text Classification

Diving into SML

SML to code frames and topics

Some work by Burscher et al., 2014 and Burscher et al., 2015

- Humans can code generic frames (human-interest, economic, ...)
- Humans can code topics from a pre-defined list
- But it is very hard to formulate an explicit rule (as in: code as 'Human Interest' if regular expression R is matched)

⇒ This is where you need supervised machine learning!

SML to code frames and topics

Some work by Burscher et al., 2014 and Burscher et al., 2015

- Humans can code generic frames (human-interest, economic, ...)
- Humans can code topics from a pre-defined list
- **But it is very hard to formulate an explicit rule**
(as in: code as 'Human Interest' if regular expression R is matched)

SML to code frames and topics

Some work by Burscher et al., 2014 and Burscher et al., 2015

- Humans can code generic frames (human-interest, economic, ...)
- Humans can code topics from a pre-defined list
- **But it is very hard to formulate an explicit rule**
(as in: code as 'Human Interest' if regular expression R is matched)

⇒ This is where you need supervised machine learning!

TABLE 4
Classification Accuracy of Frames in Sources Outside the Training Set

	VK/NRC $\rightarrow Tel$	VK/TEL $\rightarrow NRC$	NRC/TEL $\rightarrow VK$
Conflict	.69	.74	.75
Economic Cons.	.88	.86	.86
Human Interest	.69	.71	.67
Morality	.97	.90	.89

Note. VK = Volkskrant, NRC = NRC/Handelsblad, TEL = Telegraaf

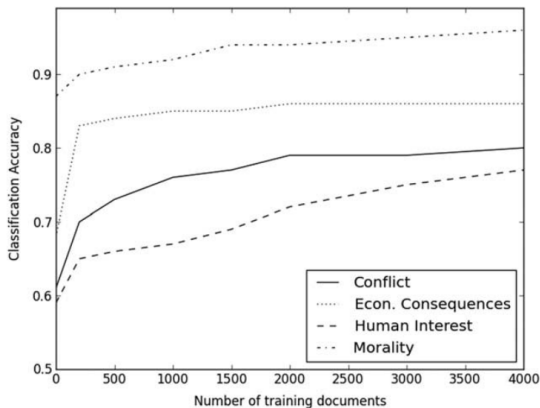


FIGURE 1 Relationship between classification accuracy and number of training documents.

FIGURE 1

Learning Curves for the Classification of News Articles and PQs

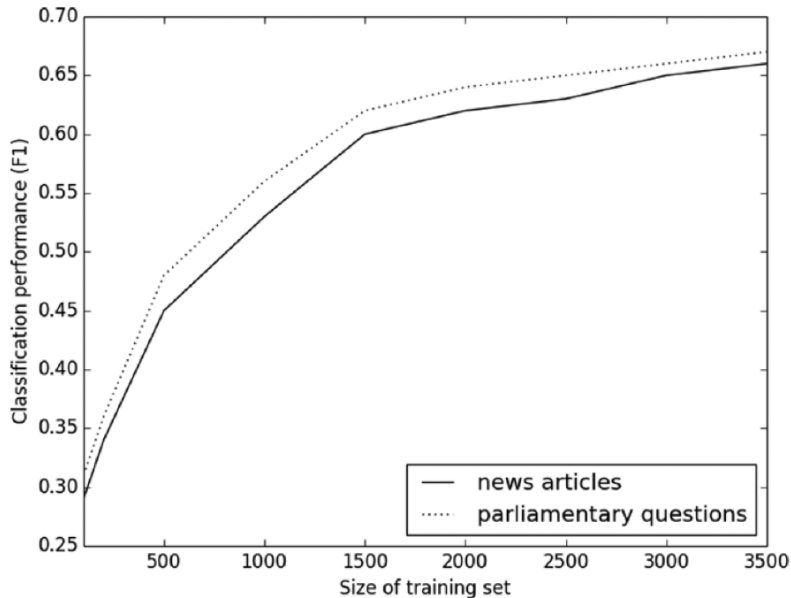


TABLE 1

F1 Scores for SML-Based Issue Coding in News Articles and PQs

Issue	News Articles			PQs	
		All Words	Lead Only		All Words
Features	N	F1	F1	N	F1
Macroeconomics	413	.54	.63	172	.46
Civil rights and minority issues	327	.34	.28	192	.53
Health	444	.70	.71	520	.81
Agriculture	114	.72	.76	159	.66
Labor and employment	217	.43	.49	174	.58
Education	188	.79	.71	229	.78
Environment	152	.34	.44	237	.59
Energy	81	.35	.59	67	.66
Immigration and integration	150	.50	.57	239	.78
Transportation	416	.58	.67	306	.81
Law and crime	1198	.70	.69	685	.77
Social welfare	115	.33	.34	214	.54
Community development and housing	113	.45	.44	136	.72
Banking, finance, and commerce	622	.62	.67	188	.58
Defense	393	.59	.55	196	.71
Science, technology, and communication	426	.64	.59	57	.53
International affairs and foreign aid	1,106	.70	.64	352	.65
Government operations	1,301	.71	.72	276	.48
Other issue	3,322	.84	.80	360	.51
Total	11,089	.71	.68	4,759	.69

NOTE: The F1 score is equal to the harmonic mean of recall and precision. Recall is the fraction of relevant documents that are retrieved, and precision is the fraction of retrieved documents that are relevant.

What does this mean for our research?

It we have 2,000 documents with manually coded frames and topics. . .

- we can use them to train a SML classifier
- which can code an unlimited number of new documents
- with an acceptable accuracy (at least for some of them)

Some easier tasks even need only 500 training documents, see Hopkins and King, 2010.

What does this mean for our research?

It we have 2,000 documents with manually coded frames and topics. . .

- we can use them to train a SML classifier
- which can code an unlimited number of new documents
- with an acceptable accuracy (at least for some of them)

Some easier tasks even need only 500 training documents, see Hopkins and King, 2010.

Supervised Machine Learning for Text Classification

An implementation

An implementation

Let's say we have a list of tuples with movie reviews and their rating:

```
1 reviews=[("This is a great movie",1),("Bad movie",-1), ... ...]
```

And a second list with an identical structure:

```
1 test=[("Not that good",-1),("Nice film",1), ... ...]
```

Both are drawn from the same population, it is pure chance whether a specific review is on the one list or the other.

Based on an example from <http://blog.dataquest.io/blog/naive-bayes-movies/>

But we can do even better

We can use different vectorizers and different classifiers.

Supervised Machine Learning for Text Classification

Classifiers

Different classifiers

Typical options in a nutshell:

- Naïve Bayes
- Logistic Regression
- Support Vector Machine (SVM/SVC)
- Random forests

Supervised Machine Learning for Text Classification

Vectorizers

Different vectorizers

1. CountVectorizer (=simple word counts)
2. TfidfVectorizer (word counts ("term frequency") weighted by number of documents in which the word occurs at all ("inverse document frequency"))

$$tfidf_{t,d} = tf_{t,d} \cdot idf_t$$

There are different ways to weigh the idf score. A common one is taking the logarithm:

$$idf_t = \log \frac{N}{n_t}$$

where N is the total number of documents and n_t is the number of documents containing term t .

1.6. Model

1.6. Model

- Preprocessing (e.g., stopwords removal)
- Remove words below a specific threshold ("occurring in less than $n = 5$ documents") \Rightarrow spelling mistakes etc.
- Remove words above a specific threshold ("occurring in more than 50% of all documents") \Rightarrow de-facto stopwords
- Not only to improve prediction, but also performance (can reduce number of features by a huge amount)

NB with Count

	precision	recall
positive reviews:	0.87	0.77
negative reviews:	0.79	0.88

NB with TfIdf

	precision	recall
positive reviews:	0.87	0.78
negative reviews:	0.80	0.88

LogReg with Count

	precision	recall
positive reviews:	0.87	0.85
negative reviews:	0.85	0.87

LogReg with TfIdf

	precision	recall
positive reviews:	0.89	0.88
negative reviews:	0.88	0.89

Summing up

Summing up

Revisiting the difference between the dictionary approach and the SML

What *is* our fitted classifier again?

Essentially, just a formula

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)}}$$

where β_0 is an intercept¹, β_1 a coefficient for the frequency (or tf·idf score) of some word, β_2 a coefficient some other word.

If our fitted *vectorizer* contains 5,000 words, we thus have 5,001 coefficients.

(for logistic regression in this case, but same argument applies to other classifiers as well)

¹Machine Learning people sometimes call the intercept “bias” (yes, I know, that’s confusing)



But isn't that then essentially very much like a dictionary, except that the words have different weights?

In some sense, yes.

- But we don't pretend that we can construct the dictionary *a priori*.
- It's specifically tailored to our use-case.
- The weights are *really* essential here.

We *could* print all coefficients-word pairs, but probably it's enough to just look at those with the largest absolute value:

ELI5

```
In [98]: import eli5
eli5.show_weights(pipe, top=10)
```

Out[98]: **y=1** top features

Weight ⁷	Feature
+9.043	great
+8.487	excellent
+6.908	perfect
... 37662 more positive ...	
... 37178 more negative ...	
-6.507	worse
-7.347	poor
-8.341	boring
-8.944	waste
-8.976	bad
-9.152	awful
-12.749	worst

```
In [111]: eli5.show_prediction(clf, test[0][0], vec=vec)
```

```
Out[111]: y=1 (probability 0.844, score 1.689) top features
```

Contribution [?]	Feature
+1.920	Highlighted in text (sum)
-0.232	<BIAS>

It is a **rare** and **fine** spectacle, an allegory of death and transfiguration that is **neither** preachy nor **mawkish**, a work of **mature** and courageous insight, northfork avoids arthouse distinction by refusing to belong to a kind. **unlike** the **most** memorable and accomplished film to impose an **obvious** comparison, **wim wenders'** 1987 wings of desire (der himmel über berlin), it sustains an ambivalence in a narrative spectrum spanning from the **mundane** to the supernatural. this story of earthly and celestial eminent domains in the **american** west withholds the **fairytale** literalness that **marked** its **german** predecessor in the **ad hoc** **genre** of angels shedding their wings with obsequious sentimentalism. its celestial transcendence, be it inspired by **doleful** faith or **impelled** by a fever **dream**, never parts **ways** with crud and rot. this firm grounding redounds to **great** credit for **writers** and **directors** **mark** and **michael** **polish**.

- 51

For instance, the negation and/or intensifier problem.

- n -grams as features

For instance, the negation and/or intensifier problem.

Possible approaches

- n -grams as features
- preprocessing (?)
- deep learning
- ...

⇒ But ultimately, it's just an empirical question how big the problem is!

Summing up

A note on the input data

The input scikit-learn expects

A training dataset consisting of:

1. an array (e.g., a list) of labels (`y_train`)
2. a corresponding array (e.g., a list) of feature vectors (`X_train`)

A test dataset consisting of:

1. an array (e.g., a list) of labels (`y_test`)
2. a corresponding array (e.g., a list) of feature vectors (`X_test`)

The feature vectors can be created via a *vectorizer*, but could in principle also just be lists themselves.

We use a lowercase `y` because it is a onedimensional vector, and an uppercase `X` because it is a two-dimensional matrix.

- ## Technology

D. $\lim_{x \rightarrow 0} \frac{f(x)}{g(x)} = \frac{f(0)}{g(0)}$ if $f(0) = g(0) = 0$ and $f'(0) \neq g'(0)$.

Summing up



Any questions?

Things to remember

- unsupervised vs supervised
- rough understanding of different techniques and when to use them
- evaluation metrics (e.g., precision, recall)

Let's do an exercise!

References



Boukes, M., van de Velde, B., Araujo, T., & Vliegenthart, R. (2020). What's the Tone? Easy Doesn't Do It: Analyzing Performance and Agreement Between Off-the-Shelf Sentiment Analysis Tools. *Communication Methods and Measures*, 14(2), 83–104.
<https://doi.org/10.1080/19312458.2019.1671966>



Boumans, J. W., & Trilling, D. (2016). Taking stock of the toolkit: An overview of relevant automated content analysis approaches and techniques for digital journalism scholars. *Digital Journalism*, 4(1), 8–23. <https://doi.org/10.1080/21670811.2015.1096598>



Burscher, B., Odijk, D., Vliegenthart, R., de Rijke, M., & de Vreese, C. H. (2014). Teaching the computer to code frames in news: Comparing two supervised machine learning approaches to frame analysis. *Communication Methods and Measures*, 8(3), 190–206.
<https://doi.org/10.1080/19312458.2014.937527>



Burscher, B., Vliegenthart, R., & De Vreese, C. H. (2015). Using supervised machine learning to code policy issues: Can classifiers generalize across contexts? *The ANNALS of the American Academy of Political and Social Science*, 659(1), 122–131.
<https://doi.org/10.1177/0002716215569441>



Hopkins, D. J., & King, G. (2010). A method of automated nonparametric content analysis for social science. *American Journal of Political Science*, 54(1), 229–247.



Hutto, C. J., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International AAAI Conference on Weblogs and Social Media*.



Pennebaker, J. W., Booth, R. J., & Francis, M. E. (2007). Linguistic Inquiry and Word Count: LIWC.



Thelwall, M., Buckley, K., & Paltoglou, G. (2012). Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1), 163–173.
<https://doi.org/10.1002/asi.21662>