



MonkeyLearn

- Text classification

Two kinds of Models

Text Classifiers

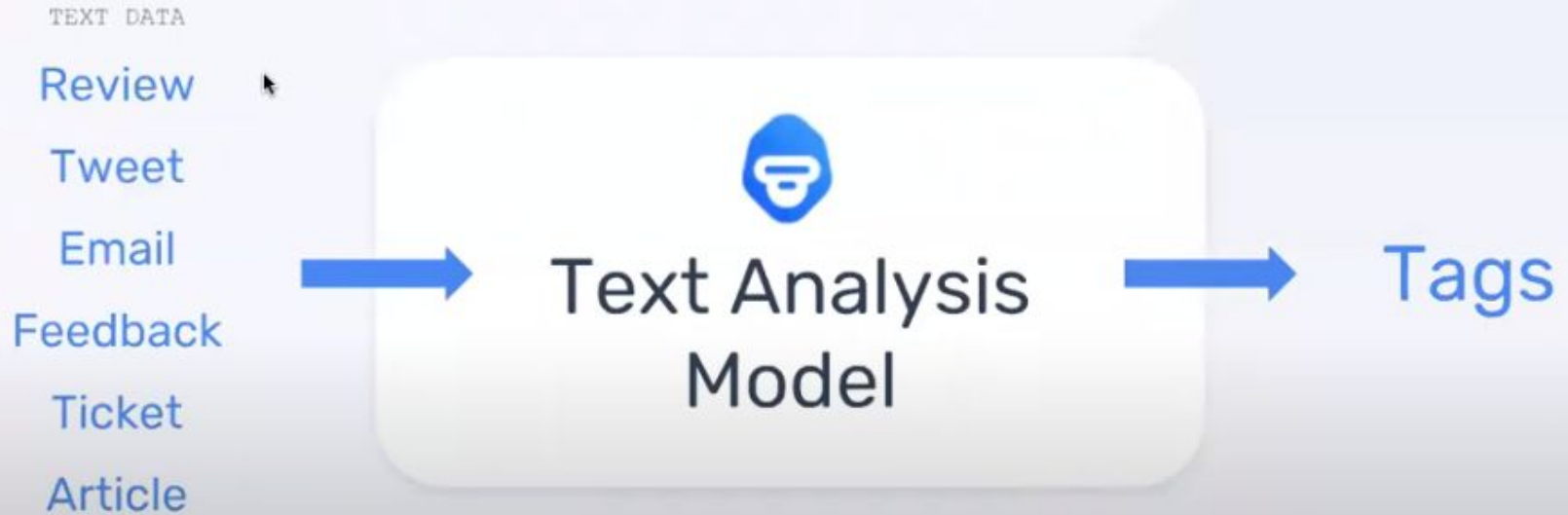
Classify text into categories or tags, e.g. sentiment, topic, urgency.

Text Extractors

Extract data from a given text, e.g. keyword, names, emails

Text Classifiers

Limitation: input data has to be CSV or Excel format



Text Classifiers

- Sentiment Analysis

Tags: positive, negative, neutral

- Urgency Detector

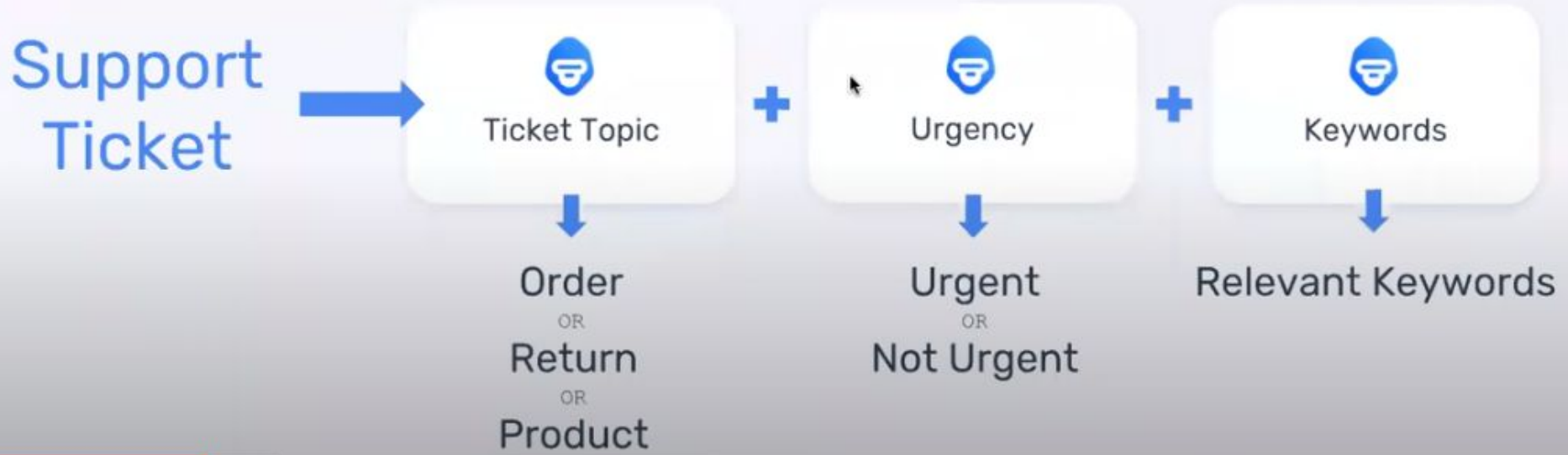
Tags: urgent, not urgent

- Topic Detector

Tags: shipment, billing, return, inquiry

Text Classifiers

Multiple Models Together



Keyword Extractor

Classifiers

Topic:

Order Issue

Sentiment:

Negative

Urgency:

Urgent

Terrible Service - I ordered a Playstation 4 and FIFA 19 from your store last week. I'm furious to find out that the tracking number doesn't work and my order might be lost. Please respond ASAP.

Extractors

Keywords:

terrible service,
store, tracking
number, order,
ASAP, etc.

Products:

Playstation 4,
FIFA 19

Train your own text classifier model

Upload data → define tags → tag data → test

TRAINING

Tag Data

Choose one or more tags that apply and click confirm. As new texts appear, the model will learn from your criteria.

Q Search data...

Destiny: The Taken K Destiny: The Taken King Collector's Edition for Xbox One When will this be available for preorder?

I've already reserved a copy with GameStop but would like to purchase through Best Buy instead if I can get some credible information on when it will be available.

TAGS

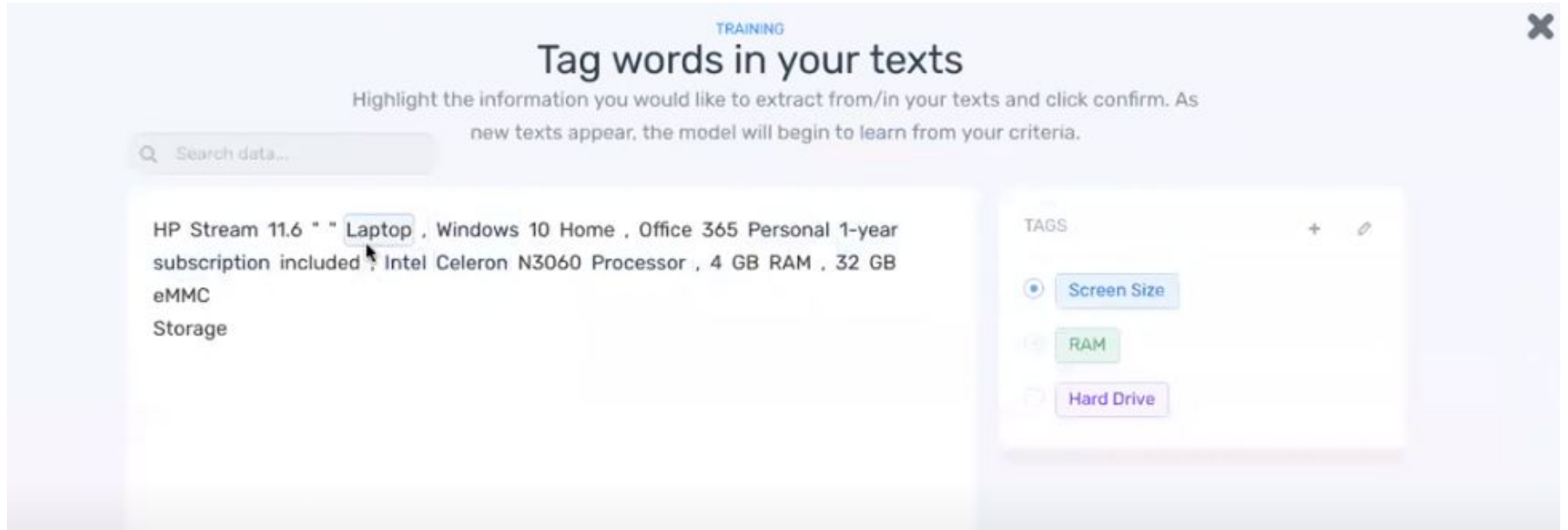
☒ Availability

☐ Order Issues

☐ Returns

Train your own keyword extractor model

Upload data → define tags → tag data → test



The screenshot shows a web interface for training a keyword extractor model. At the top, the word 'TRAINING' is in small blue letters. Below it, the main heading is 'Tag words in your texts'. A sub-instruction reads: 'Highlight the information you would like to extract from/in your texts and click confirm. As new texts appear, the model will begin to learn from your criteria.' On the left, there is a search bar with a magnifying glass icon and the text 'Search data...'. Below the search bar is a text area containing the following text: 'HP Stream 11.6 " " Laptop , Windows 10 Home , Office 365 Personal 1-year subscription included , Intel Celeron N3060 Processor , 4 GB RAM , 32 GB eMMC Storage'. The word 'Laptop' is highlighted with a blue box, and a mouse cursor is pointing at it. On the right, there is a panel titled 'TAGS' with a plus sign and an edit icon. It contains three tags: 'Screen Size' (selected with a blue radio button), 'RAM' (with a green radio button), and 'Hard Drive' (with a purple radio button).

TRAINING

Tag words in your texts

Highlight the information you would like to extract from/in your texts and click confirm. As new texts appear, the model will begin to learn from your criteria.

Search data...

HP Stream 11.6 " " Laptop , Windows 10 Home , Office 365 Personal 1-year subscription included , Intel Celeron N3060 Processor , 4 GB RAM , 32 GB eMMC Storage

TAGS

- ☒ Screen Size
- ☐ RAM
- ☐ Hard Drive

Supported algorithms

- Naive Bayes
- SVM (Support Vector Machines)

Naive Bayes

- Probabilistic algorithm
 - Calculate the probability of each tag for a given text, and then output the tag with the highest one
- Laplace smoothing

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

Text	Tag
"A great game"	Sports
"The election was over"	Not sports
"Very clean match"	Sports
"A clean but forgettable game"	Sports
"It was a close election"	Not sports

$$P(sports|a \text{ very close game}) = \frac{P(a \text{ very close game}|sports) \times P(sports)}{P(a \text{ very close game})}$$

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

$$P(a|Sports) \times P(very|Sports) \times 0 \times P(game|Sports)$$

Laplace smoothing:

Word	P (word Sports)	P (word Not Sports)
a	$(2 + 1) \div (11 + 14)$	$(1 + 1) \div (9 + 14)$
very	$(1 + 1) \div (11 + 14)$	$(0 + 1) \div (9 + 14)$
close	$(0 + 1) \div (11 + 14)$	$(1 + 1) \div (9 + 14)$
game	$(2 + 1) \div (11 + 14)$	$(0 + 1) \div (9 + 14)$

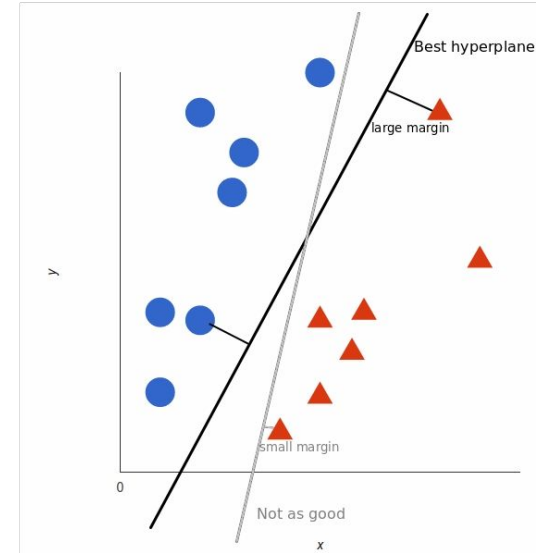
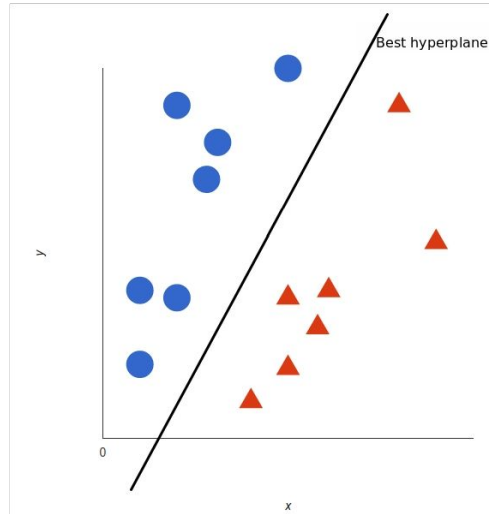
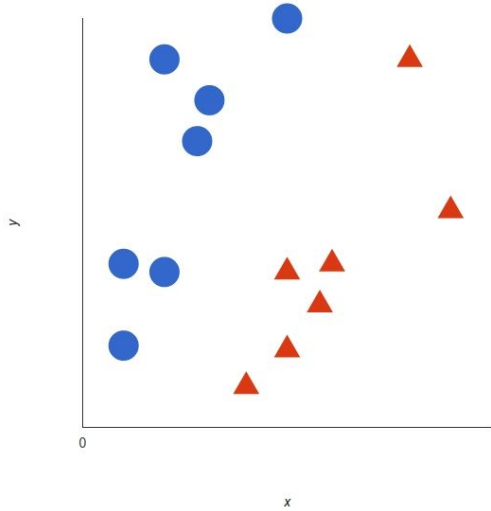
$$\begin{aligned}
 &P(a|Sports) \times P(very|Sports) \times P(close|Sports) \times P(game|Sports) \times \\
 &P(Sports) \\
 &= 2.76 \times 10^{-5} \\
 &= \underline{0.0000276}
 \end{aligned}$$

Classified as sport!

$$\begin{aligned}
 &P(a|Not Sports) \times P(very|Not Sports) \times P(close|Not Sports) \times P(game|Not Sports) \times \\
 &P(Not Sports) \\
 &= 0.572 \times 10^{-5} \\
 &= \underline{0.00000572}
 \end{aligned}$$

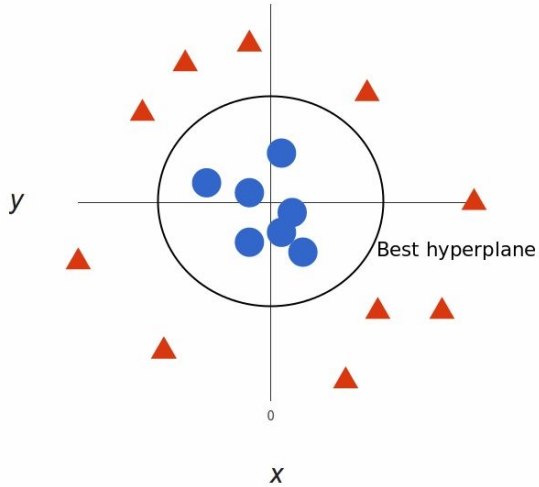
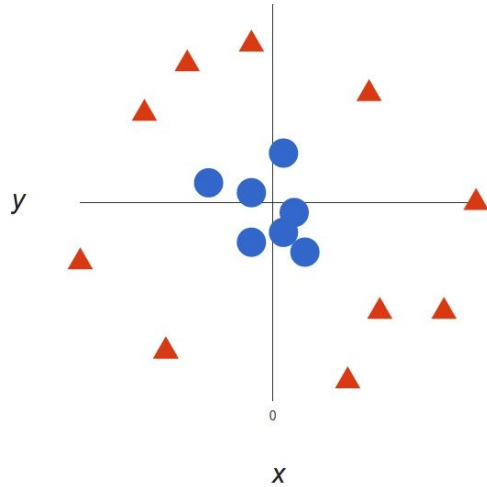
SVM (Support Vector Machines)

- Linear data



SVM (Support Vector Machines)

- Nonlinear data



References

- Text classifiers & text extractors
 - <https://www.youtube.com/watch?v=vHSadS-1wOA>
- Naive Bayes
 - <https://monkeylearn.com/text-classification-naive-bayes/>
 - <https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/>
- SVM
 - <https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/>
 - <https://monkeylearn.com/text-classification-support-vector-machines-svm/>