1.------

There are basically 2 broad types of algorithms for sentiment analysis

**Lexicon based VS Learning based techniques**

* ***Lexicon based techniques*** use a dictionary to perform entity-level sentiment analysis. This technique uses dictionaries of words annotated with their semantic orientation (polarity and strength) and calculates a score for the polarity of the document. Usually this method gives high precision but low recall.
* ***Learning based techniques*** require creating a model by training the classifier with labeled examples. This means that you must first gather a dataset with examples for positive, negative and neutral classes, extract the features/words from the examples and then train the algorithm based on the examples.

Choosing which method you will use heavily depends on the application, domain and language. Using lexicon based techniques with large dictionaries enables us to achieve very good results. Nevertheless they require using a lexicon, something which is not always available in all languages. On the other hand Learning based techniques deliver good results nevertheless they require obtaining datasets and require training.

Machine learning based techniques are fancier  and popular now a days .

As I have used those so I will briefly discuss those.For lexicon driven approach you can refer to [https://www.aclweb.org/anthology...](https://www.aclweb.org/anthology/J/J11/J11-2001.pdf)

Sentiment Analysis will require the following pre-processing:

1. ***Noise Removal*** - Cleaning the data from irrelevant news as well as advertisements/bio(if you have collected data by web crawling)

2. **Domain Classification** - Categorizing the data to different domains - "Markets",  "Economy", "Industry", "Technology" and so on. It is as necessary as the algorithm because you will have different set of features for different domains and thus, each domain should have different classifier. For example, A positive news in Technology sector for Microsoft may be a negative news for Apple stocks.

There is no single algorithm that performs well in all topics/domains/applications. Be prepared to see that the accuracy of your classifier can be as high as 90% in one domain/topic and as low as 60% in some other.

For example you might find that *Max Entropy with Chi-square* as feature selection ([https://cmm.cit.nih.gov/maxent/l...](https://cmm.cit.nih.gov/maxent/letsgo.html) ) is the best combination for restaurant reviews, while for twitter the*Binarized Naïve Bayes with Mutual Information*feature ([Machine Learning Tutorial: The Naive Bayes Text Classifier](http://blog.datumbox.com/machine-learning-tutorial-the-naive-bayes-text-classifier/) ) selection outperforms even the SVMs. Be prepared to see lots of weird results. Particularly in case of twitter, avoid using lexicon based techniques because users are known to use idioms, jargons and twitter slangs what heavily affect the polarity of the tweet.

3.**Feature Selection** - In learning based techniques, before training the classifier, you must select the words/features that you will use on your model. You can’t just use all the words  because there are several irrelevant words within them.

The features can be unigrams and/or bigrams or higher ngrams with/without punctuation and with/without stopwords.

*Some of the current features used are :*

* **Terms presence and frequency**: These features are individual words or word n-grams and their frequency counts. It either gives the words binary weighting (zero if the word appears, or one if otherwise) or uses term frequency weights to indicate the relative importance of features .
* **Parts of speech (POS)**: Finding adjectives, as they are important indicators of opinions.
* **Opinion words and phrases**: These are words commonly used to express opinions including good or bad, like or hate. On the other hand, some phrases express opinions without using opinion words. For example: cost me an arm and a leg.
* **Negations**: The appearance of negative words may change the opinion orientation like not good is equivalent to bad.

4. **Classification Algorithm** - The text classification methods using ML approach can be roughly divided into supervised and unsupervised learning methods. The supervised methods make use of a large number of labeled training documents. The unsupervised methods are used when it is difficult to find these labeled training documents.

Some of the classifiers along with the libraries (as building classifiers from scratch is painstakingly difficult :P ) that you can try are these:

* **Naive bayes** - BernoulliNB, GaussianNB, MultinomialNB. [Naive Bayes](http://scikit-learn.org/stable/modules/naive_bayes.html)
* **Support Vector Classifiers** - LinearSVC, PolynomialSVC, RbfSVC, NuSVC. [Support Vector Machines](http://scikit-learn.org/stable/modules/svm.html)
* **Maximum Entropy Model** - GIS, IIS, MEGAM, TADM [nltk.classify package](http://www.nltk.org/api/nltk.classify.html)

**NOTE: *Different Classifiers deliver different results***

Make sure you try as many classification methods as possible. Have in mind that different algorithms deliver different results. Also note that some classifiers might work better with specific feature selection configuration.

Generally it is expected that state of the art classification techniques such as SVM would outperform more simple techniques such as Naïve Bayes. Nevertheless be prepared to see the opposite. Sometimes Naïve Bayes is able to provide the same or even better results than more advanced methods. Don’t eliminate a classification model only due to its reputation.

I'll point you to a few good resources(both for extracting features/learning techniques):-

1. ***Google Word2Vec***(<https://code.google.com/p/word2vec/>) : Provides methods to convert text to features automatically. Here's a link to a tutorial notebook([Bag of Words Meets Bags of Popcorn](http://www.kaggle.com/c/word2vec-nlp-tutorial/forums/t/11189/notebook-of-the-tutorial))

2. ***Deep Learning(***[***Deeply Moving: Deep Learning for Sentiment Analysis***](http://nlp.stanford.edu/sentiment/)***)***: This is currently the model used by Stanford.

3. ***Sentiment Analysis competition(***[***Sentiment Analysis on Movie Reviews***](http://www.kaggle.com/c/sentiment-analysis-on-movie-reviews)***):***Recently held Kaggle competition on the same - you can find some pretty cool ideas implemented here - best of all, most of the code is available

**DATASET** for training and testing : [Sentiment Analysis](http://ai.stanford.edu/~amaas/data/sentiment/)

Some of the papers on sentiment analysis may help you -

One of the earlier works by Bo Pang, Lillian Lee [http://acl.ldc.upenn.edu/acl2002...](http://acl.ldc.upenn.edu/acl2002/EMNLP/pdfs/EMNLP219.pdf)

A comprehensive survey of sentiment analysis techniques [http://www.cse.iitb.ac.in/~pb/cs...](http://www.cse.iitb.ac.in/~pb/cs626-449-2009/prev-years-other-things-nlp/sentiment-analysis-opinion-mining-pang-lee-omsa-published.pdf)

Study by Hang Cui, V Mittal, M Datar using 6-grams [http://citeseerx.ist.psu.edu/vie...](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.83.5942&rep=rep1&type=pdf)

Sources:

[10 Tips for Sentiment Analysis projects](http://blog.datumbox.com/10-tips-for-sentiment-analysis-projects/)

[Sentiment analysis algorithms and applications: A survey](http://www.sciencedirect.com/science/article/pii/S2090447914000550)