Investment KPI Extraction

Team Members:

Sindhuja Kasula (801076683)
Vrushali Mahuli (801076699)
Madhuri Pawle (801083244)
Anagha Sarmalkar (801077504)

Problem Statement:

Extraction of Investor's KPI's and a brief description of KPI usage from a document, given it has investment related KPI in it.

For this we decided to divide the project into 2 main parts:

1) Classify the documents into "Investment related" and "Not Investment related"
2) Extract the KPI's from the classified documents

Data Available:

We have 704 business articles(papers) in pdf format; which include KPI's related to customers, investment and employees.

Data Cleaning:

The given data come in PDF format which could not immediately be edited and analyzed.

- We converted the pdfs to text files using PDFMiner.
- We removed the stopwords, converted the text into lower case and stemmed every word in this text for better training of the classifier using nltk.
- Porter Stemmer was used for stemming.
- Out of 704 files we successfully converted 698 files into text files. (There was error with encodings of the remaining files due to non-english characters)

Why PDFMiner was used?

PDFMiner is a tool for extracting information from PDF documents. Unlike other PDF-related tools, it focuses entirely on getting and analyzing text data. PDFMiner allows one to obtain the exact location of text in a page, as well as other information such as fonts or lines. It includes a PDF converter that can transform PDF files into other text formats (such as HTML). It has an extensible PDF parser that can be used for other purposes than text analysis

- We tried parsing with PyPDF and PDFMiner.
- PyPDF could not parse some versions of the pdf.
- PDFMiner could however parse 99.14% of the documents which gave us more documents to work with.

Why NLTK was used?

The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP).

It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning. It also includes graphical demonstrations and sample data sets as well as accompanied by a cook book and a book which explains the principles behind the underlying language processing tasks that NLTK supports.
We utilized NLTK package for the following:

- We converted the text into lower case for better string matching using regex and removed the stopwords since they provide unwanted bias during modelling. English stop words library from NLTK was leveraged.
- We stemmed the words to their morphological variant to improve retrieval effectiveness and to reduce the size of indexing files. This was done using the Porter Stemmer from NLTK

Solution:

Part 1: Classifying the documents:

Steps:

- We first used google to find out words and phrases that define Investment KPI's; we then listed them in a txt file.
- We checked if there existed a word/ phrase in the document that matched with the phrases in the KPI list.
- If a word/phrase was found; we added the document to a folder named '1' (this folder contained the "Investment Related" documents). The documents were converted to txt format when saving in the folder.
- If nothing was found the document was added to the folder '0' in txt format.
- While storing these txt files, the words were stemmed, and the stopwords were removed.
- After parsing all the documents, we had a set of labelled data with labels 0 and 1
- This data was then vectorized using tf-idf vectorizer and count-vectorizer for comparison.
- The data was then split into 2 for training and testing purposes. 80% of the data was reserved for training.

- The training data was then trained using Multinomial Naive Bayes, Linear SVM and Logistic Regression for both the methods of Vectorization (A total of 6 models). The corresponding observations have been recorded.

Handling the ambiguous KPI words:

- We created a list of ambiguous words that are important for Investment related KPI's.
- The sentences related to these KPI's were added to the excel only after making sure that at least 1 non-ambiguous KPI was present in the document.

Why we used a supervised learning model:

We tried clustering using the k-means classifier(unsupervised learning). The results differed with the slightest variation in iterations. The documents were classified on different factors. Therefore, supervised learning seemed like the best bet for classification.

Why Naive Bayes was used:

- Naive Bayes Algorithm is a fast, highly scalable algorithm.
- It is a simple algorithm that depends on doing a bunch of counts.
- Great choice for Text Classification problems. It's a popular choice for spam email classification.
- It can be easily train on small dataset.

Why SVM was used:

- SVM's are very good when we have no idea on the data.
- Works well with even unstructured and semi structured data like text, Images and trees.
- The kernel trick is real strength of SVM. With an appropriate kernel function, we can solve any complex problem.
- SVM models have generalization in practice, the risk of overfitting is less in SVM.

Why Logistic Regression was used:

- Logistic Regression is one of the most used Machine Learning algorithms for binary classification.
- It is a widely used technique because it is very efficient, does not require too many computational resources.
- it's highly interpretable, it doesn't require input features to be scaled, it doesn't require any tuning, it's easy to regularize, and it outputs well-calibrated predicted probabilities.

Part 2: Extracting KPI's from the documents.

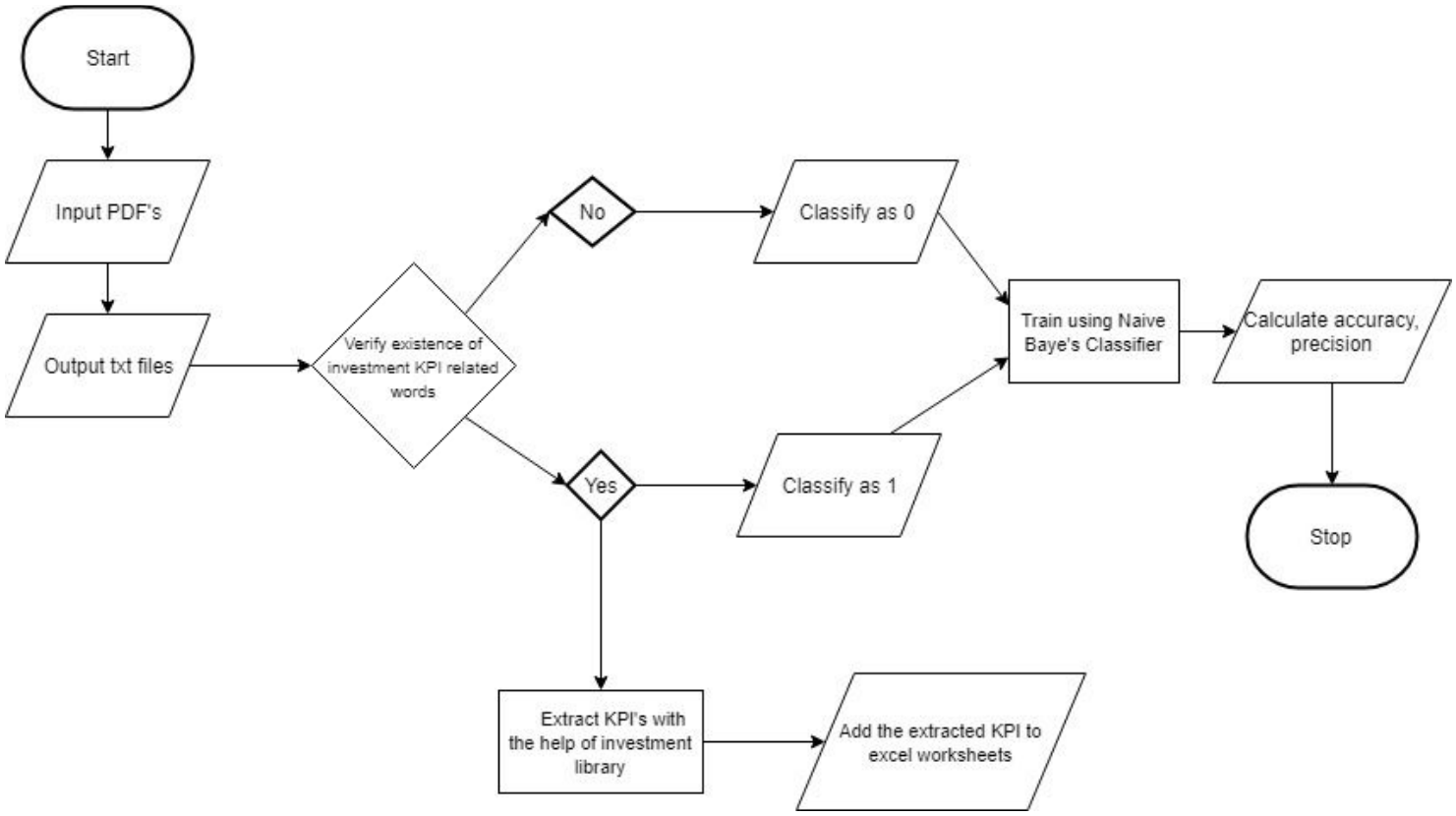
Steps:

- We created an Excel file for storing the KPI's
- When the documents were classified as "Investment Related", the parser searched for the sentences in the document that contained the words in the KPI list. This was done using regex.
- A new sheet in excel was created where sheet name was renamed to the document name i.e Each sheet in the KPI excel contained KPI's from one particular file.
- The KPI name was added to the cells; and all the sentences extracted related to that KPI from the document were added to the adjacent cell.

Example of words and phrases in the KPI list:

Net profit margin
Operating cash flow (OCF)
Payment error rate
Internal audit cycle time
Finance error report
Debt to equity ratio
Return on equity
Accounts receivable turnover

Flowchart of the Process:



Results:

PDFs Parsed : 698/704 documents parsed

Task 1: Extraction of KPI's

Here are some of the sample KPI's that were extracted from the pdf's:

- "The Gallup national poll (conducted March 2003 **margin** of error 5%), showed that 80% of Americans generally favour setting higher emissions and pollution standards for business and industry."
- "We found that environmental reputation is positively related to the objective financial performance measures in terms of ROA (b = 1.44, P < 0.001), ROI (b = 1.43, P < 0.001), profit **margin** (b = 1.31, P < 0.001) and EPS (b = 1.34,P < 0.001)."
- "This resulted in Vanke's share valuation dropping from 25 RMB to 19.58 RMB in just 1 day."
- "Julian and Ofori-Dankwa (2013) studied a firm in a sub-Saharan African emerging economy and found that the firm's return on sales, **return on equity**, and net **profitability** were negatively related to CSR expenditures."

How do we know it's correct?

We first defined what we accept to be an Investment Related document.

- Any document that explains/compares the effect of any policy/changes with the profitability.
- Any document that explains the effects of various parameters on share and market value of a company
- We used random sampling to manually analyze a set of 70 documents from the labelled dataset created (10% of the converted pdf's). The observations in the form of Confusion Matrix are as follows.

		Actual Values	
		Negative	Positive
Predicted Values	Negative	28	7
	Positive	6	29

- The metrics using randomly sampled labelled data

  Accuracy: **0.8143**
  Precision: **0.8286**
  Recall: **0.8056**

Specificity: **0.8235**
F1 score: **0.8169**

Task 2: Classification model

Preliminary comparison results:
Accuracy when using CountVectorizer with Naive Baye's: **~75%**
Accuracy when using tf-idfwith Naive Baye's: **~70%**
Accuracy when using CountVectorizer with SVM: **~81%**
Accuracy when using tf-idf with SVM: **~80%**
Accuracy when using CountVectorizer with Logistic Regression: **~80%**
Accuracy when using tf-idf with Logistic Regression: **~82%**

Metrics of Logistic Regression classifier using TF-IDF Vectorizer:

		Actual Values	
		Negative	Positive
Predicted Values	Negative	85	14
	Positive	13	28

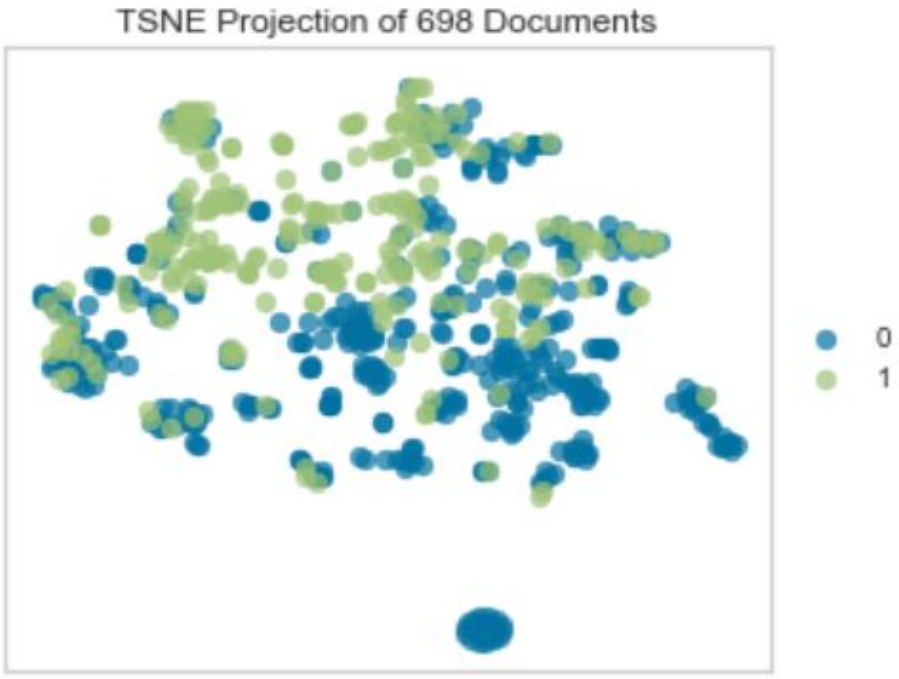
Accuracy: **0.8071**
Precision: **0.8586**
Recall: **0.8673**
Specificity: **0.6667**
F1 score: **0.8629**

Conclusion:

We conclude that Logistic Regression classifier with tf-idf vectorizer is the best option in this case.

Visualizations:

Visualization of labelled data using T-SNE:
This visualization gives an idea about the distribution of the dataset. A lot of the documents are similar.



Error Analysis and Future Scope:

- The errors in parsing were due to encodings in a different language.
- The documents are a lot similar and the training dataset is small to build a complete classifier.
- KPI may be found in Titles or References, which is also counted in this model; this can be improved.

What we learned in the process:

- Different types of text classifiers and their advantages.
- We learned to use different summarizers(though we did not use it in the end)
- Learned to use T-SNE visualizer
- Learned to work with Excels in Python

How to Run the Code:

The project is divided into 2 python notebooks. You can run them in that order:

1) The Data_Process_Extract: This is the first part where the pdf files are parsed and text from them is extracted and stored in txt format.
   This also extracts the KPI and stores in excel.
2) The Classifier: This file has the implementation of different supervised learning classifiers that we build for comparison.

References:

http://dataaspirant.com/2017/02/06/naive-bayes-classifier-machine-learning/
https://statinfer.com/204-6-8-svm-advantages-disadvantages-applications/
https://github.com/iamiamn/nlp-chinese_text_classification/blob/master/categorizing.py