Sentiment analysis of financial news headlines

1. About the data

The chosen dataset contains data on financial news. To be exact it consists of two columns: the first one has financial news headlines there; the second one represents the sentiment of a given headline, it can be negative, neutral, or positive. This dataset contains 4845 rows.

The dataset can be found on Kaggle.com, at this link:
https://www.kaggle.com/datasets/ankurzing/sentiment-analysis-for-financial-news

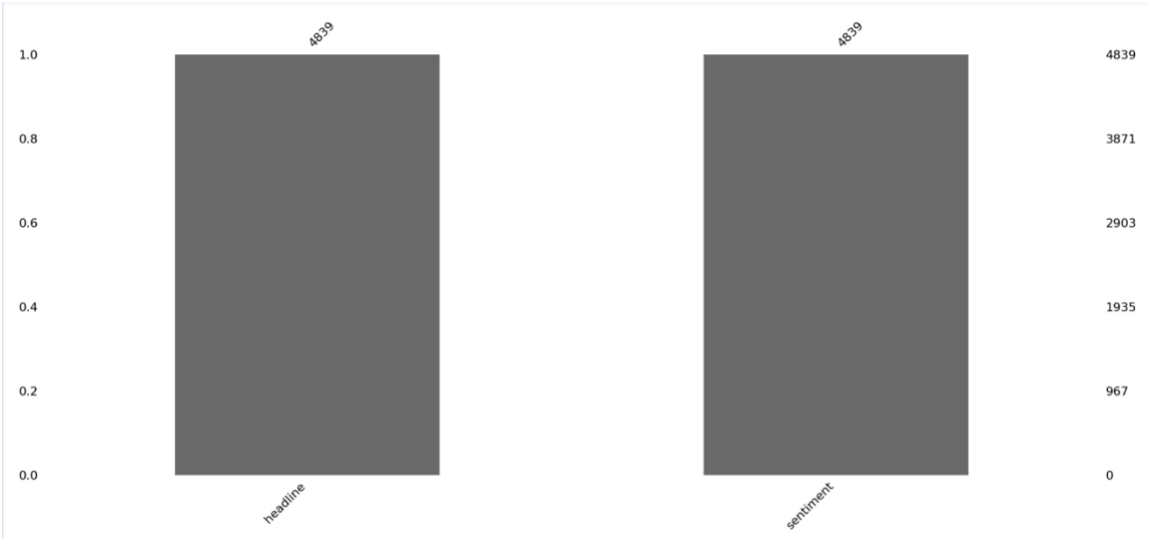
2. Data preprocessing and cleaning

After loading the dataset and printing out the first couple of rows I noticed that the names of the columns do not make sense, so renaming them was the first thing I did. As a result, there are two columns:

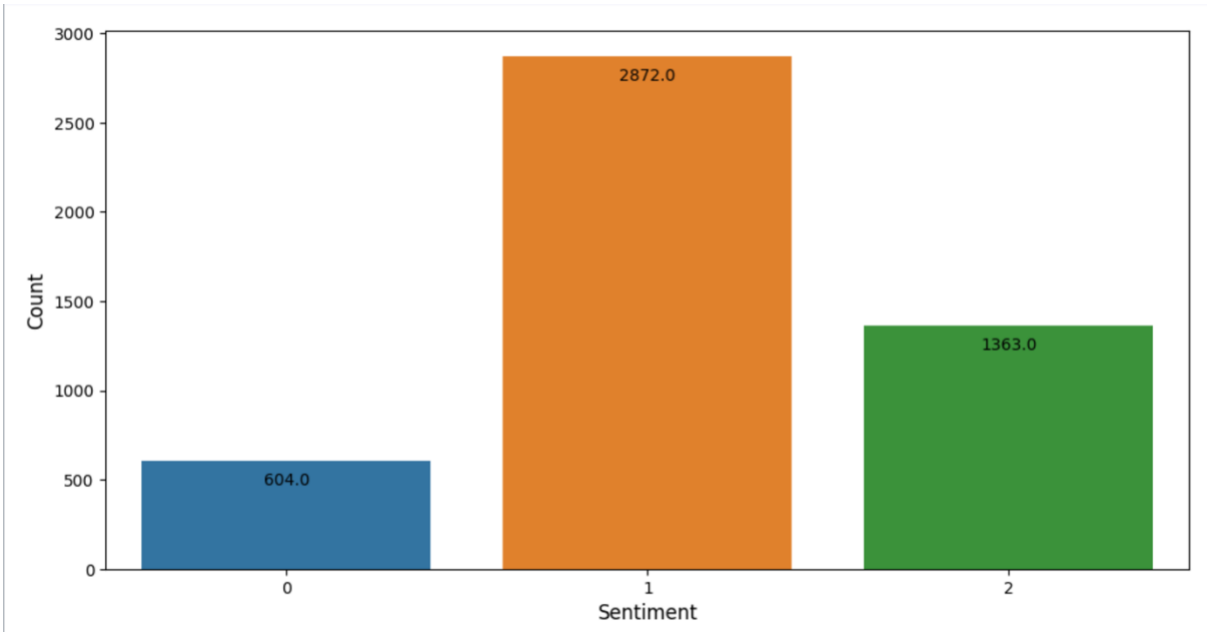
- headline – news headline
- sentiment – the sentiment of a specific news headline (with values “negative”, “neutral” and “positive”)

As the next step, I also changed the categorical variable “sentiment” into a numerical one, now negative sentiment is represented by the number 0, neutral – by 1, and positive – by 2

Now, with the data looking good I start with the data cleaning. The first thing I did was look if there were any missing values in this dataset, but fortunately, there were none, so there was no need to perform any missing value treatment.
Then I checked if there are any duplicates in the dataset. Turns out there’re 6 duplicated rows, so I deleted them all except for the first one. After that, I made sure that the duplicates were deleted properly by plotting a graph that shows the number of entries in each column. As we can see from the graph below there are 4839 rows in this dataset now.



In the next step, I decided to see which sentiment category is the biggest by creating a bar chart to make the differences more obvious. The graph below makes it clear that the biggest (2872) sentiment category is 1, which corresponds to “neutral”, positive sentiment is around 2 times less common (1364 entries), and negative sentiment is the least represented category with 604 entries.



3. Text preprocessing

Text preprocessing is a crucial step for sentiment analysis, it prepares the data and changes it into a format suitable for its later analysis.
The first and easy step was lowercasing, after that, I separated the headlines into tokens/words – performed tokenization – using nltk library in python and its word_tokenize function. Then I removed the punctuation and stop words since they don’t have any significant meaning and just create more noise in the data. I chose to perform lemmatization after that because it takes context into consideration and transforms words into their lemmas. As the final step of text preprocessing, I removed special characters and numbers, since they were not relevant to the sentiment analysis.

4. Classification task

The primary task of this work is sentiment analysis, where a classifier is created, that allows financial news to be classified into a suitable sentiment group using its headline.

First, I transformed the preprocessed data into numerical features using TF-IDF vectorization. After that I created a classifier using LinearSVS (Linear Support Vector Classifier) algorithm and then I trained this classifier on the vectorized data to predict the sentiment of financial news headlines.

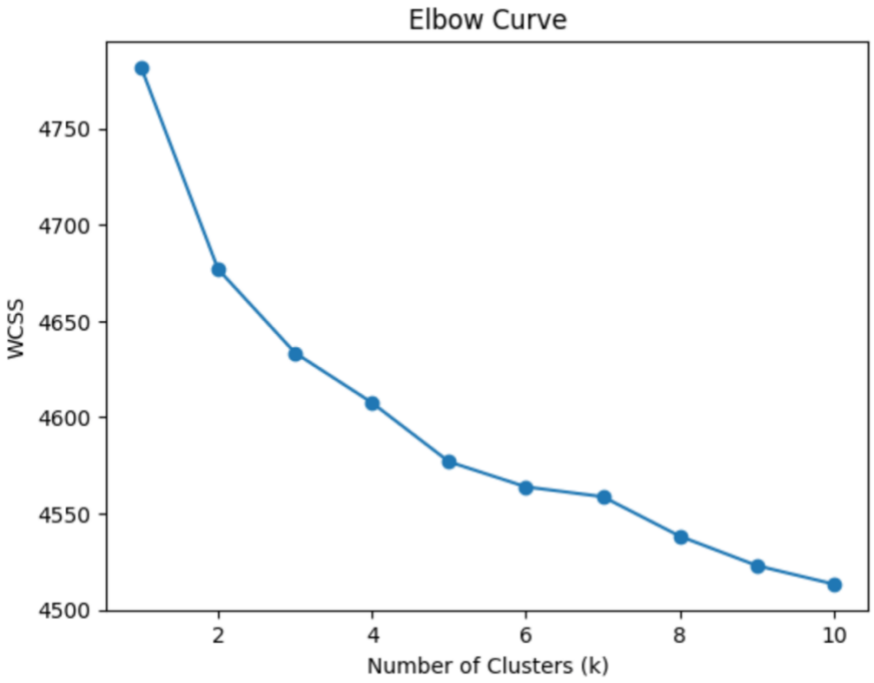
The results of the classifier's predictions were evaluated using different measures, such as accuracy, recall, and F1 score. The classifier best performed in predicting neutral sentiment

data with a precision of 0.78, recall – 0.89, and F1 score – 0.83. The precision of negative sentiment predictions was second best(0.75), but its recall was the worst(0.51), which means that 75% of predicted negative headlines were actually negative, but only 51% of all of the actual negative sentiments were predicted as negative. The positive sentiment had the worst precision(0.67) but the second-best recall(0.57). Both negative and positive sentiment categories had the same F1 score of 0.61. The overall accuracy of the model turned out to be 0.75, which is quite a good score.

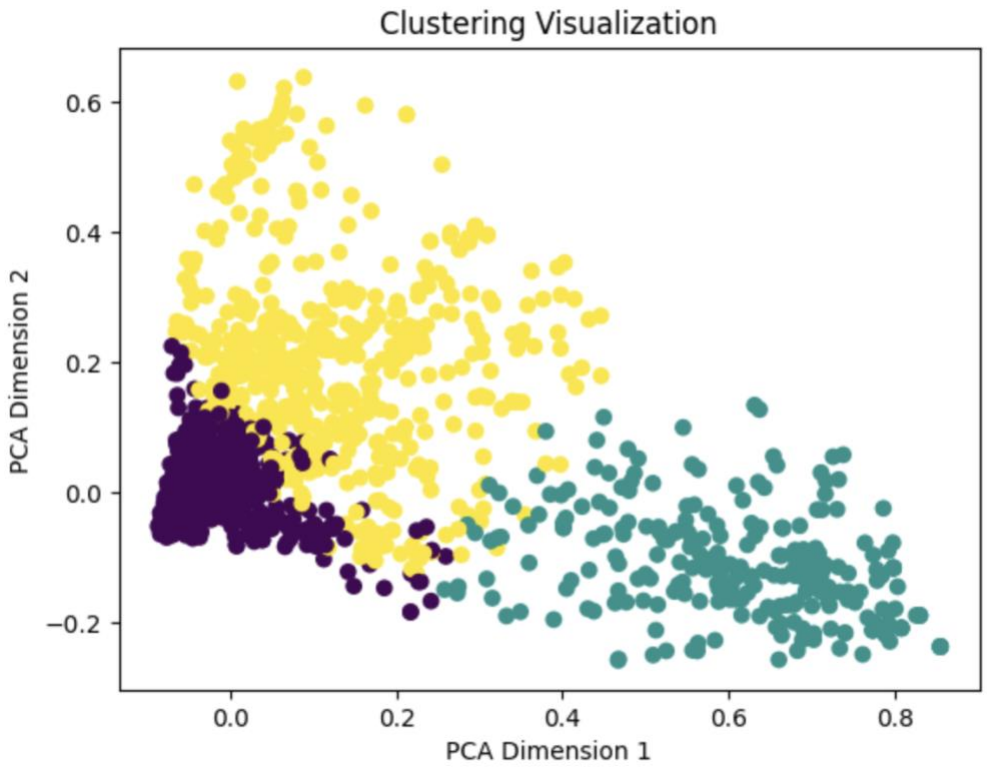
```
Accuracy: 0.7486225895316805
              precision    recall  f1-score   support

           0       0.75      0.51      0.61       189
           1       0.78      0.89      0.83       855
           2       0.67      0.57      0.61       408

    accuracy                           0.75      1452
   macro avg       0.73      0.65      0.68      1452
weighted avg       0.74      0.75      0.74      1452
```

5. Clustering

For the clustering task, I used the K-Means clustering algorithm to group the news headlines based on the similarities they might have. I performed the dimensionality reduction using PCA as well. Prior to creating the clusters I used an elbow method to find the optimal amount of clusters, so I chose 3 as my number of clusters. I checked if it is correct by changing the number of clusters a couple of times and building a scatter plot and the clusters on it turned out to be more separated and looked like clusters when there were just 3 of them.



Then I created a scatter plot with 3 clusters. Each point on this plot represents a news headline and its color represents the cluster it is assigned to. Here we can see that the separations of the clusters in this model is not very good, what we will see later in a silhouette score as well.



To evaluate the model I used two scores: calinski-harabasz and silhouette, both giving different insights. So the silhouette score turned out to be quite small as I predicted before is 0.01. The silhouette score ranges from -1 to 1. A score close to 0 suggests that the sample is near the decision boundary between two neighboring clusters or may be assigned to the wrong cluster.

The calinski-harabasz score however showed quite a good result of 77.4. It measures the ratio of between-cluster dispersion to within-cluster dispersion, with higher scores indicating better-defined and more separated clusters.

6. Collocation analysis

Collocation analysis is used to identify word combinations that are the most meaningful within the given text. These word combinations usually mean specific things that wouldn't usually be obvious by themselves.
As the first step here I made a list of all separate words in this dataset. The next step was to identify the bigram collocations, after that, I ranked them by their usage using the likelihood ratio. Below is the list of the top ten collocations that I got as a result. There are a lot of variations of "million euros", which makes sense considering that these headlines are for financial news and it also seems like a lot of these headlines are about Finland.

```
Collocations:
eur mn
net sale
operating profit
eur million
corresponding period
euro mln
mln euro
oyj hel
omx helsinki
stora enso
```

7. Extra

After the collocation analysis, I got curious to see what are the top used words for positive and negative sentiment groups. By creating the subsets of data and using a wordcloud package I ended up getting these 2 word clouds, for positive sentiment:

For negative sentiment:

These seems to prove the idea that the headlines in this dataset are mostly about Finland.