

Let's predict the sentiment!

SENTIMENT ANALYSIS IN PYTHON

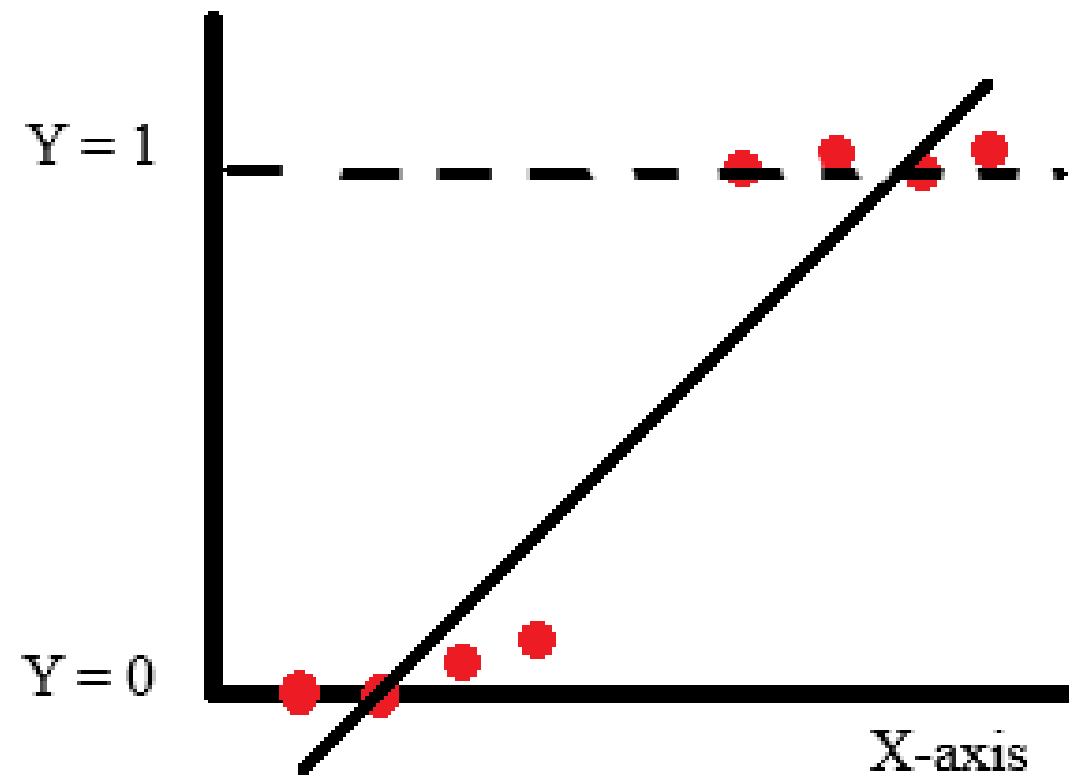


Violeta Misheva
Data Scientist

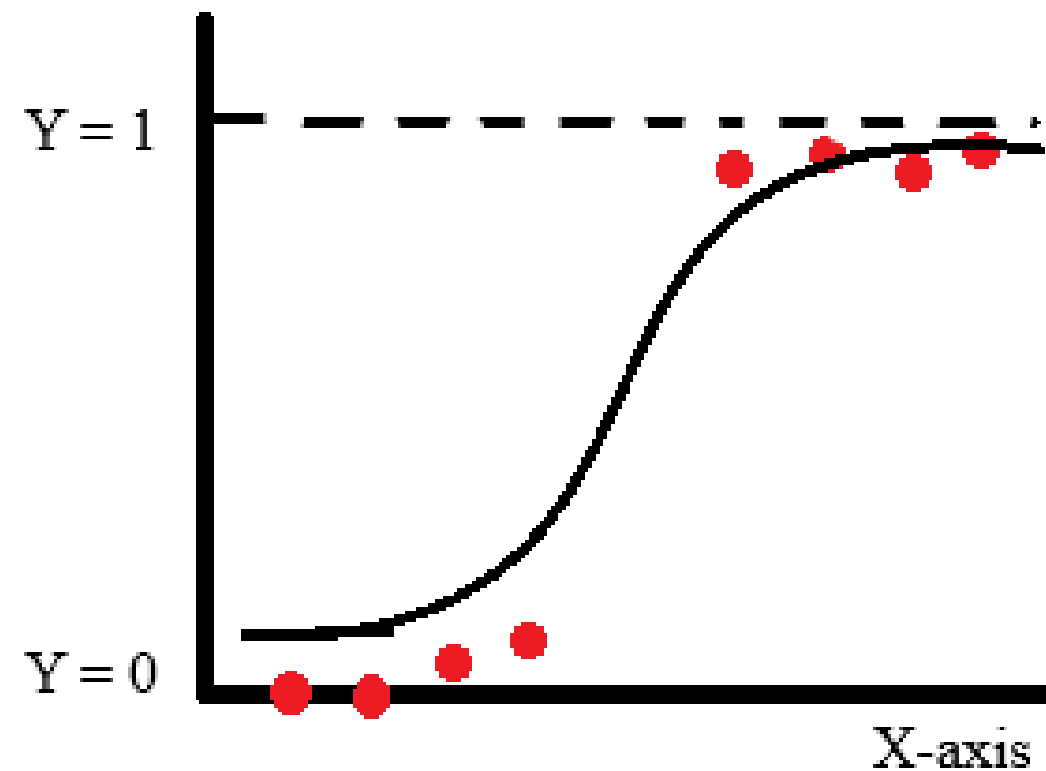
Classification problems

- Product and movie reviews: positive or negative sentiment (binary classification)
- Tweets about airline companies: positive, neutral and negative (multi-class classification)

Linear and logistic regressions



Linear regression

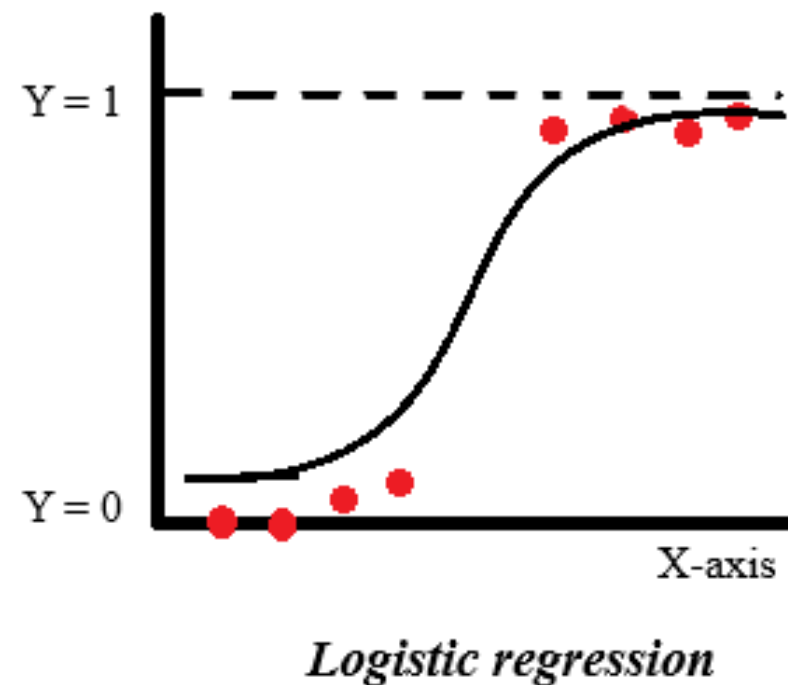


Logistic regression

Logistic function

- Linear regression: numeric outcome
- Logistic regression: probability:

$$Probability(sentiment = positive|review)$$



Logistic regression in Python

```
from sklearn.linear_model import LogisticRegression
```

```
log_reg = LogisticRegression().fit(X, y)
```

Measuring model performance

Accuracy: Fraction of predictions our model got right.

- The higher and closer the accuracy is to 1, the better

```
# Accuracy using score  
score = log_reg.score(X, y)  
print(score)
```

```
0.9009
```

Using accuracy score

```
# Accuracy using accuracy_score  
from sklearn.metrics import accuracy_score
```

```
y_predicted = log_reg.predict(X)  
accuracy = accuracy_score(y, y_predicted)
```

```
0.9009
```

Let's practice!
SENTIMENT ANALYSIS IN PYTHON

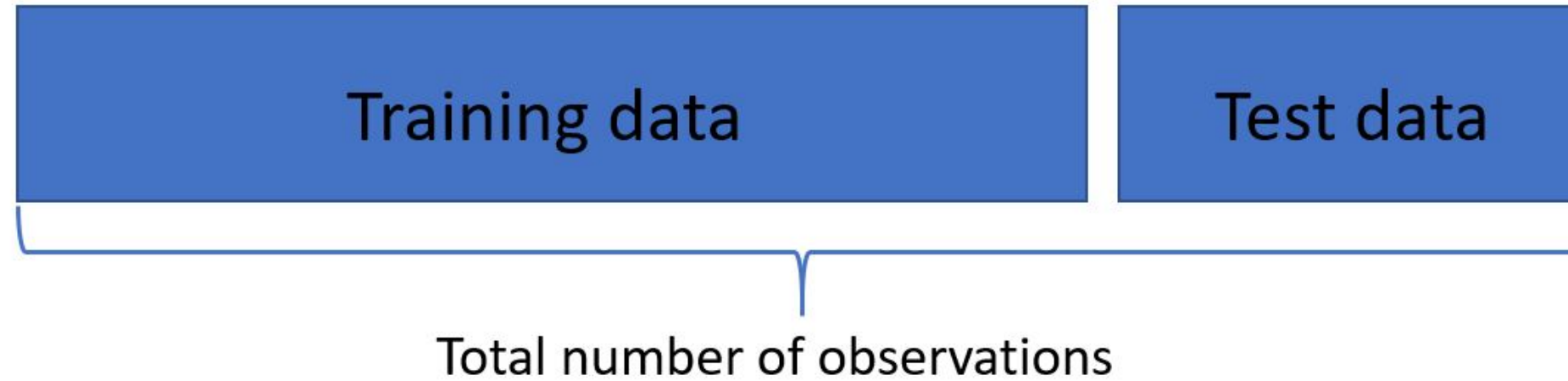
Did we really predict the sentiment well?

SENTIMENT ANALYSIS IN PYTHON



Violeta Misheva
Data Scientist

Train/test split



- Training set: used to train the model (70-80% of the whole data)
- Testing set: used to evaluate the performance of the model

Train/test in Python

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=123, stratify=
```

- **X** : features
- **y** : labels
- **test_size**: proportion of data used in testing
- **random_state**: seed generator used to make the split
- **stratify**: proportion of classes in the sample produced will be the same as the proportion of values provided to this parameter

Logistic regression with train/test split

```
log_reg = LogisticRegression().fit(X_train, y_train)
```

```
print('Accuracy on training data: ', log_reg.score(X_train, y_train))
```

```
0.76
```

```
print('Accuracy on testing data: ', log_reg.score(X_test, y_test))
```

```
0.73
```

Accuracy score with train/test split

```
from sklearn.metrics import accuracy_score
```

```
log_reg = LogisticRegression().fit(X_train, y_train)
```

```
y_predicted = log_reg.predict(X_test)  
print('Accuracy score on test data: ', accuracy_score(y_test, y_predicted))
```

```
0.73
```

Confusion matrix

	Actual = 1	Actual = 0
Predicted = 1	True positive	False positive
Predicted = 0	False negative	True negative

Confusion matrix in Python

```
from sklearn.metrics import confusion_matrix
```

```
log_reg = LogisticRegression().fit(X_train, y_train)  
y_predicted = log_reg.predict(X_test)
```

```
print(confusion_matrix(y_test, y_predicted)/len(y_test))
```

```
[[0.3788 0.1224]  
 [0.1352 0.3636]]
```

Let's practice!
SENTIMENT ANALYSIS IN PYTHON

Logistic regression: revisted

SENTIMENT ANALYSIS IN PYTHON



Violeta Misheva
Data Scientist

Complex models and regularization

Complex models:

- Complex model that captures the noise in the data (overfitting)
- Having a large number of features or parameters

Regularization:

- A way to simplify and ensure we have a less complex model

Regularization in a logistic regression

```
from sklearn.linear_model import LogisticRegression
```

```
# Regularization arguments  
LogisticRegression(penalty='l2', C=1.0)
```

- L2: shrinks all coefficients towards zero
- High values of C: low penalization, model fits the training data well.
- Low values of C: high penalization, model less flexible.

Predicting a probability vs. predicting a class

```
log_reg = LogisticRegression.fit(X_train, y_train)
```

```
# Predict labels  
y_predicted = log_reg.predict(X_test)
```

```
# Predict probability  
y_probab = log_reg.predict_proba(X_test)
```

Predicting a probability vs. predicting a class

```
y_probab  
array([[0.5002245, 0.4997755],  
       [0.4900345, 0.5099655],  
       ...,  
       [0.7040499, 0.2959501]])
```

```
# Select the probabilities of class 1  
y_probab = log_reg.predict_proba(X_test)[:, 1]
```

```
array([0.4997755, 0.5099655 ..., 0.2959501])
```

Model metrics with predicted probabilities

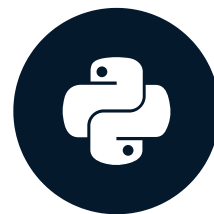
- Raise **ValueError** when applied with probabilities.
- Accuracy score and confusion matrix work with classes.

```
# Default probability encoding:  
# If probability >= 0.5, then class 1 Else class 0
```

Let's practice!
SENTIMENT ANALYSIS IN PYTHON

Bringing it all together

SENTIMENT ANALYSIS IN PYTHON



Violeta Misheva
Data Scientist

The Sentiment Analysis problem

Sentiment analysis as the process of understanding the opinion of an author about a subject

- Movie reviews
- Amazon product reviews
- Twitter airline sentiment
- Various emotionally charged literary examples

Exploration of the reviews

- Basic information about size of reviews
- Word clouds
- Features for the length of reviews: number of words, number of sentences
- Feature detecting the language of a review

Numeric transformations of sentiment-carrying columns

- Bag-of-words
- Tfidf vectorization

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
```

```
# Vectorizer syntax  
vect = CountVectorizer().fit(data.text_column)  
X = vect.transform(data.text_column)
```

Arguments of the vectorizers

- stop words: non-informative, frequently occurring words
- n-gram range: use phrases not only single words
- control size of vocabulary: max_features, max_df, min_df
- capturing a pattern of tokens: remove digits or certain characters

Important but NOT arguments to the vectorizers

- lemmas and stems

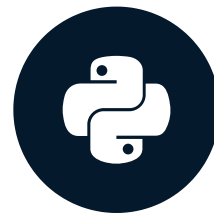
Supervised learning model

- Logistic regression classifier to predict the sentiment
- Evaluated with accuracy and confusion matrix
- Importance of train/test split

Let's practice!
SENTIMENT ANALYSIS IN PYTHON

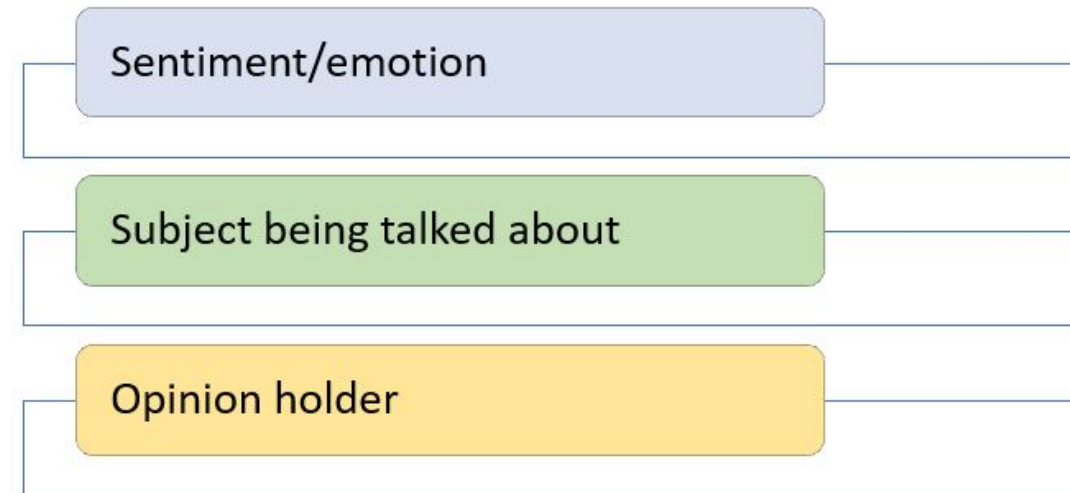
Wrap up

SENTIMENT ANALYSIS IN PYTHON

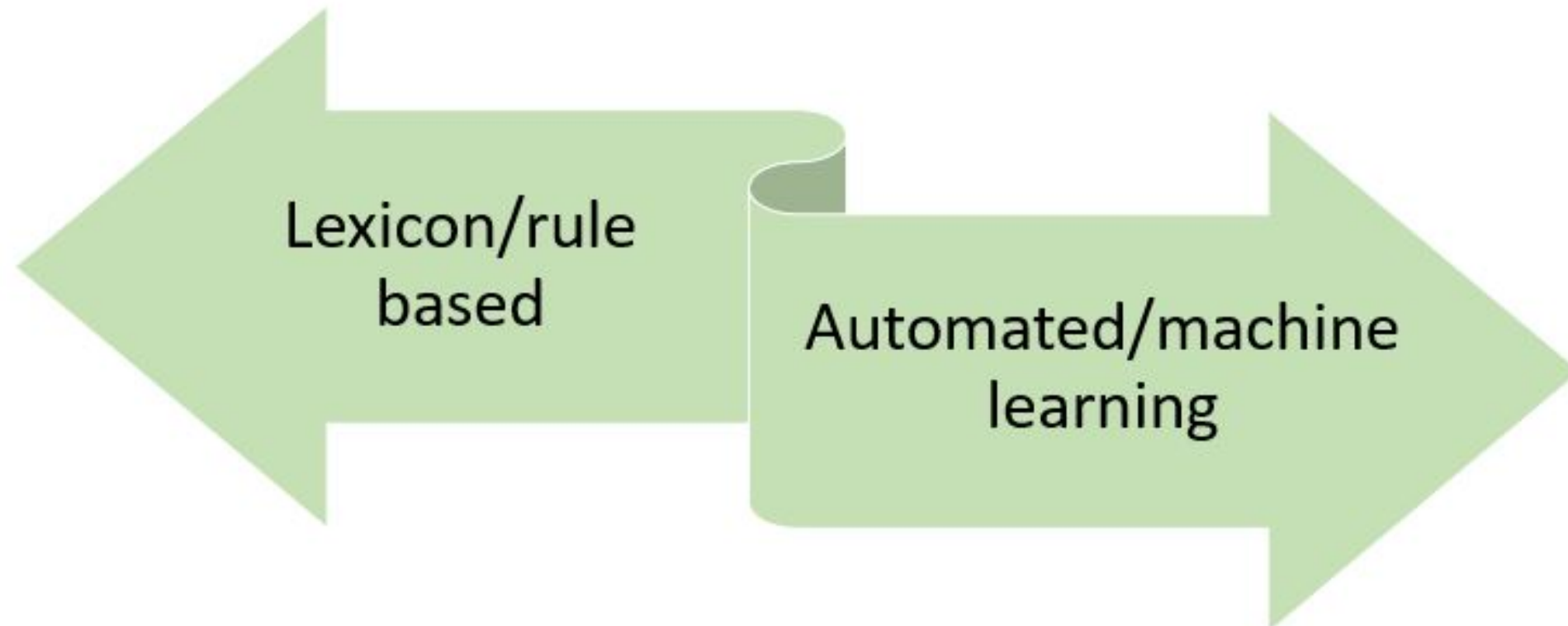


Violeta Misheva
Data Scientist

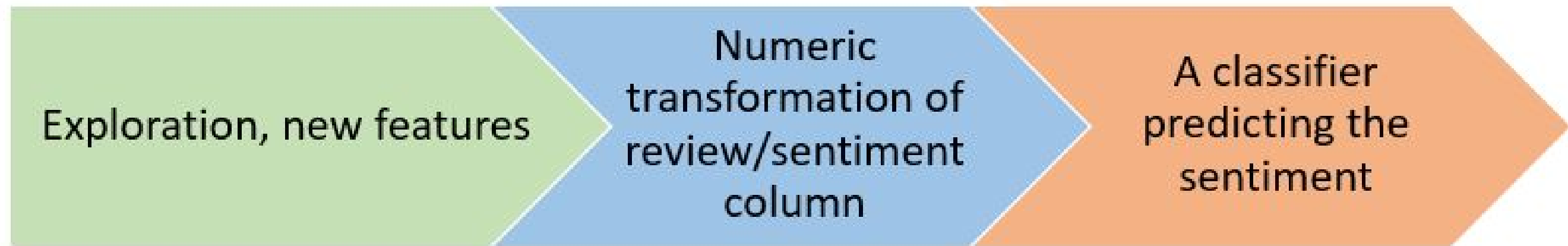
The Sentiment Analysis world



Sentiment analysis types



The automated sentiment analysis system



Congratulations!

SENTIMENT ANALYSIS IN PYTHON