Case 3. Patient Drug Review Analysis

Neural Networks for Machine Learning Applications 2023

Sakari Lukkarinen

Metropolia University of Applied Sciences

Contents

- Case 3 – Patient Drug review (Text)
  - Explorative data anaysis
  - Sentiment analysis
- Tensorflow classifier example
  - Case 3. First classification experiment
  - Next steps
- Cohen's Kappa - metrics

Case 3. Patient Drug Review

Aim is

- to predict the patient's rating for the drug (=output)
- based on patient's review (text) (=input)

Data from Drugs.com

- 200,000 patient drug reviews
  - drugName - Name of drug (e.g. Levonorgestrel, Nexaplon, …)
  - condition - Name of condition (e.g. birth control, depression, pain, …)
  - Patient review (string) - patient's review about the drug for specific condition
  - Rating (1..10) – patient's rating for the drug
  - Date – date of review
  - UsefulCount – number of users who found review useful

Drugs.com
Know more. Be sure.

Drugs A-Z | Pill Identifier | Interactions Checker | New Drugs | Pro Edition | More | Register | Sign In

Find Drugs & Conditions

Enter drug name or medical condition, pill imprint, etc.

Trending searches: gabapentin, amlodipine, lisinopril, tramadol, prednisone

Drugs & Medications | Pill Identifier | Interactions Checker | Side Effects

Browse Drugs | Browse Conditions

A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z
0-9 Advanced Search

Browse A-Z: Drug, Treatment, Condition or Class

Browse by Site Section

- Drugs A-Z
- Side Effects Checker
- Dosage Guidelines
- Manage your Meds
- Mobile Apps
- Health Professionals
- Medical News
- FDA Alerts
- New Drugs
- More

Alvesco

Generic Name: ciclesonide (inhalation) (sye KLES oh nide)
Brand Names: *Alvesco HFA*

Medically reviewed by Sophia Entringer, PharmD Last updated on Nov 26, 2019.

What is Alvesco?

Alvesco (ciclesonide) is a man-made corticosteroid. It prevents the release of substances in the body that cause inflammation.

Alvesco is used to prevent asthma attacks in adults and children who are at least 12 years. When used regularly, as prescribed by your health care provider, it will help to prevent and control symptoms of asthma.

Alvesco may also be used for purposes not listed in this medication guide.

Important information

Alvesco inhalation will not work fast enough to treat an asthma attack. Use only a fast acting inhalation medicine for an asthma attack. Tell your doctor if it seems like your asthma medications don't work as well.

Steroid medication can weaken your immune system, making it easier for you to get an infection. Steroids can also worsen an infection you already have, or reactivate an infection you recently had. Before taking Alvesco, tell your doctor about any illness or infection you have had within the past several weeks.

DRUG STATUS

Availability
Prescription only

Pregnancy & Lactation
Risk data available

CSA Schedule*
Not a controlled drug

Approval History
Drug history at FDA

NEWS Price Hikes Have Patients Turning to Craigslist for Insulin, Asthma Inhalers

Manufacturer

Sunovion Pharmaceuticals Inc.

Drug Class

Inhaled corticosteroids

Related Drugs

prednisone, Symbicort, Ventolin, Breo Ellipta, Ventolin HFA, Dulera, Atrovent, Xopenex, Nucala

User Reviews & Ratings

Alvesco reviews

7.7 / 10 20 Reviews

https://www.drugs.com/alvesco.html

Drug reviews - Alvesco

User Reviews for Alvesco

The following information is NOT intended to endorse any particular medication. While these reviews might be helpful, they are not a substitute for the expertise, skill, knowledge and judgement of healthcare practitioners.

Overview | Side Effects | Dosage | Professional | Interactions | More

Filter by condition:

--- all conditions ---

Condition	Avg. Ratings	Reviews	Compare
Asthma, Maintenance	8.2	13 reviews	123 medications
Asthma	6.9	7 reviews	145 medications
Summary of Alvesco reviews	7.7	20 reviews	

Reviews may be moderated or edited before publication to correct grammar and spelling or to remove inappropriate language and content. Reviews that appear to be created by parties with a vested interest in the medication will not be published. As reviews and ratings are subjective and self-reported, this information should not be used as the basis for any statistical analysis or scientific studies.

Share your Experience | Ask a Question

Reviews for Alvesco

Most Recent

Shellyscorner · Taken for 5 to 10 years — January 25, 2020

For Asthma, Maintenance "I've been on Alvesco longer than any of the many other steroid inhalers that I've used. In all the YEARS that I've been using it, I've never developed a thrush infection. ALL the others that I've ever used I would develop thrush fairly regularly. Even at the dose of 2 puffs twice daily, I've only developed thrush once with Alvesco. And as a maintenance med, it's worked great for me!"

9.0

What this helpful? Yes No — 1 · Report

MVE · Taken for 1 to 2 years — January 12, 2020

For Asthma, Maintenance "I've been using a different steroid inhaler for several years. My doctor changed my prescription to Alvesco. Even after the first use I noticed a difference in my asthma. I've had zero side effects and I've been using it for approximately a year."

10

What this helpful? Yes No — 1 · Report

https://www.drugs.com/comments/ciclesonide/alvesco.html

Dataset – Kaggle Winter 2018 Hackathon

https://www.kaggle.com/jessicali9530/kuc-hackathon-winter-2018

Explorative data analysis

What can we learn from the data?

Example - Team NDL: Algorithms and illnesses

https://www.kaggle.com/neilash/team-ndl-algorithms-and-illnesses

Drug Ratings Dataset: Preliminary Data Exploration

Our ideas for preliminary exploration:

- Most common conditions
- Overall best and worst reviewed drugs
- The curability of each disease
- Best drugs for each condition
- Most useful reviews
- Usefulness vs review score
- Bias in reviews
    - Users tend to review things they really liked or really disliked, fewer reviews in the middle

https://www.kaggle.com/neilash/team-ndl-algorithms-and-illnesses

Importing libraries

In [1]:

```
# ALL imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import style; style.use('ggplot')
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import time
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
```

Reading the datasets

In [2]:

```
# Create dataframes train and test
train = pd.read_csv('../input/drugsComTrain_raw.csv')
test = pd.read_csv('../input/drugsComTest_raw.csv')
```

In [3]:

```
train.head()
```

Out[3]:

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	20-May-12	27
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8	27-Apr-10	192
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5	14-Dec-09	17
3	138000	Ortho Evra	Birth Control	"This is my first time using any form of birth...	8	3-Nov-15	10
4	35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has completely turned my life around...	9	27-Nov-16	37

Check the column names and dataset sizes

In [6]:

```
list(train)
```

Out[6]:

```
['uniqueID',
 'drugName',
 'condition',
 'review',
 'rating',
 'date',
 'usefulCount']
```

In [7]:

```
train.values.shape[0], test.values.shape[0], train.values.shape[0] / test.values.shape[0]
```

Out[7]:

```
(161297, 53766, 2.999981400885318)
```

What are the most common (medical) conditions?

Common Conditions

In [10]:

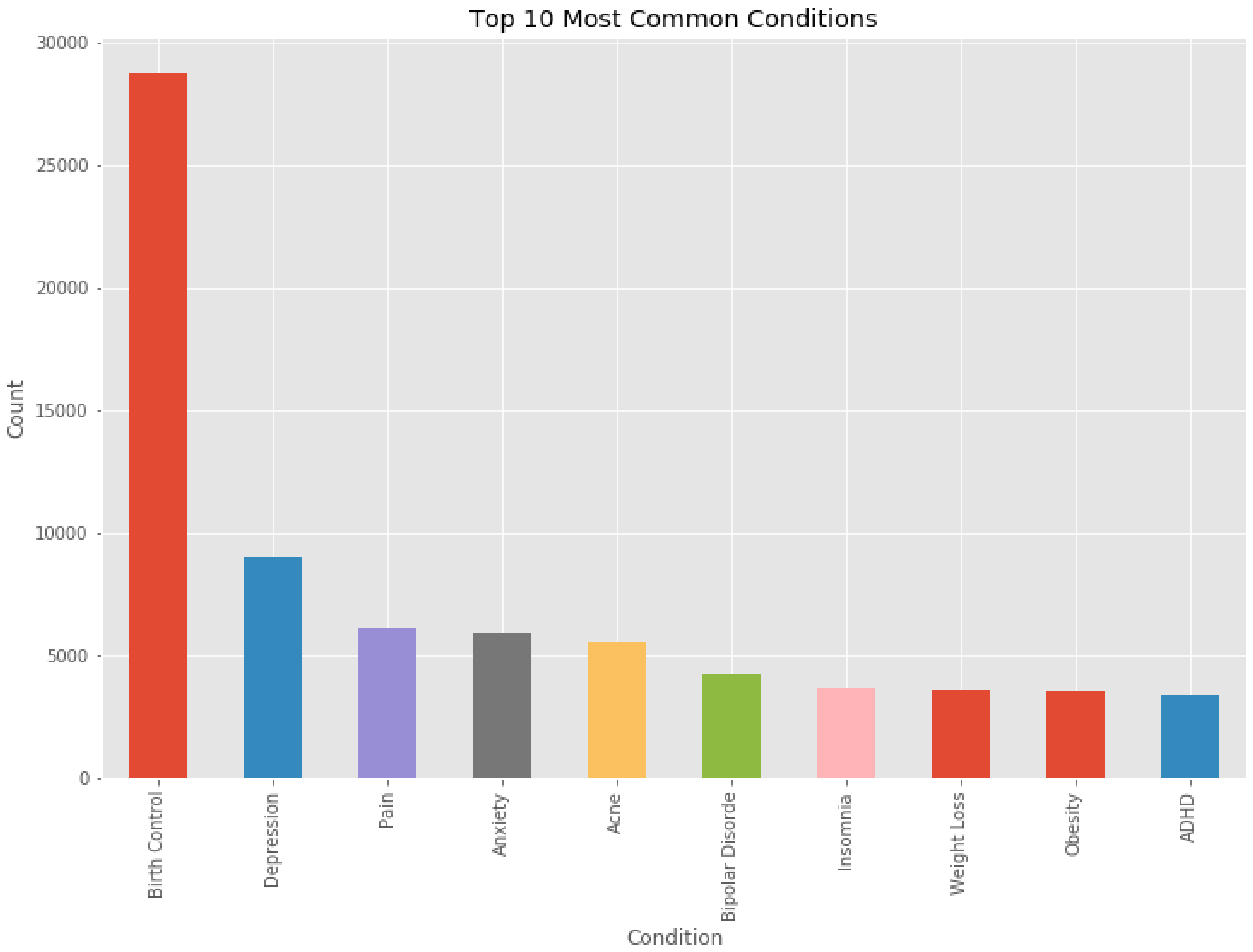
```
# I previously did this by creating and sorting a dictionary -- here's an easier way with pandas!
(Inspiration from Sayan Goswami)
conditions = train.condition.value_counts().sort_values(ascending=False)
conditions[:10]
```

Out[10]:

```
Birth Control      28788
Depression          9069
Pain                6145
Anxiety             5904
Acne                5588
Bipolar Disorde     4224
Insomnia            3673
Weight Loss         3609
Obesity             3568
ADHD                3383
Name: condition, dtype: int64
```

In [12]:

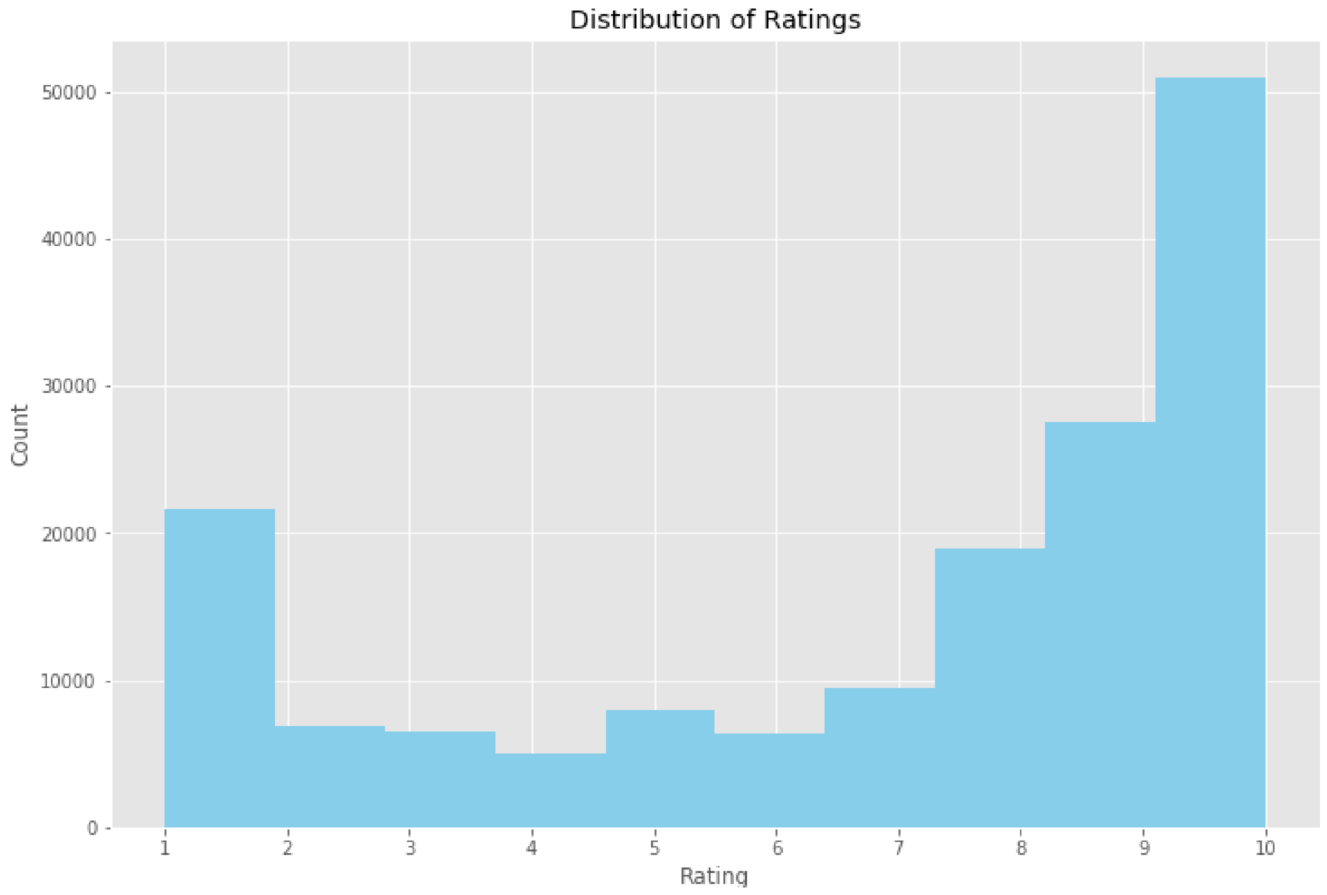
```
conditions[:10].plot(kind='bar')
plt.title('Top 10 Most Common Conditions')
plt.xlabel('Condition')
plt.ylabel('Count');
```



What is the rating distribution?

In [13]:

```
# Look at bias in review (also shown on 'Data' page in competition: distribution of ratings)
train.rating.hist(color='skyblue')
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks([i for i in range(1, 11)]);
```



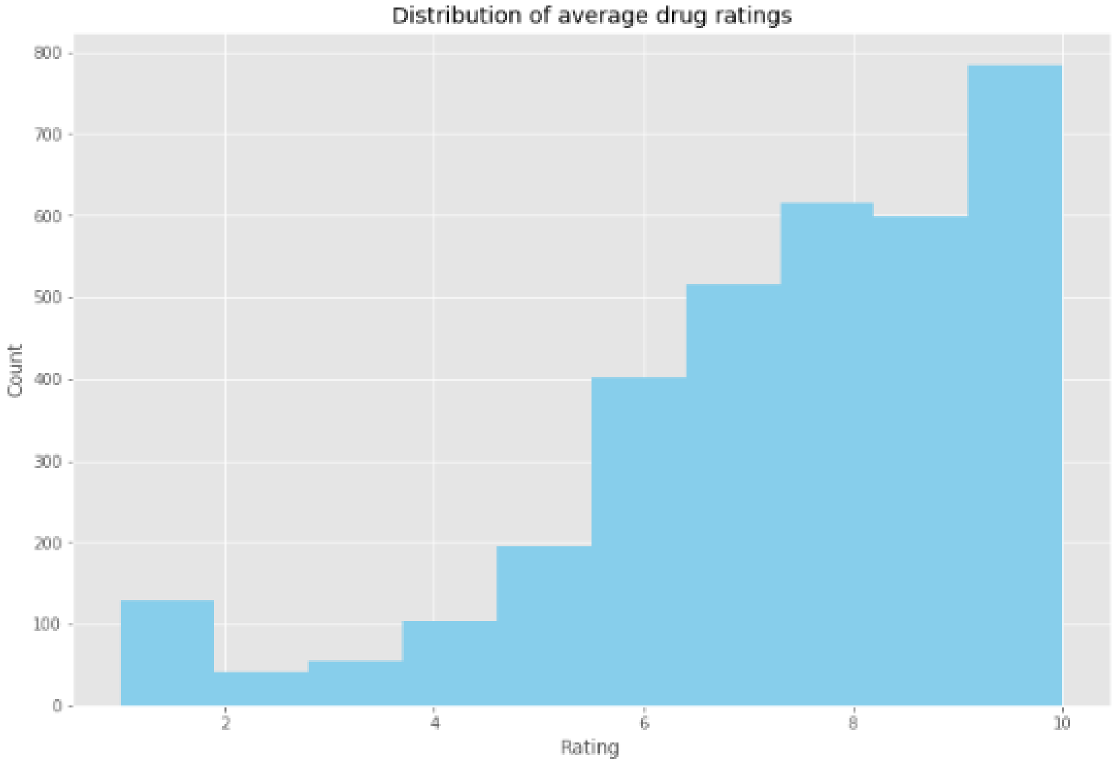
What is the average drug rating?

In [14]:

```
rating_avgs = (train['rating'].groupby(train['drugName']).mean())
rating_avgs.hist(color='skyblue')
plt.title('Distribution of average drug ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
```

Out[14]:

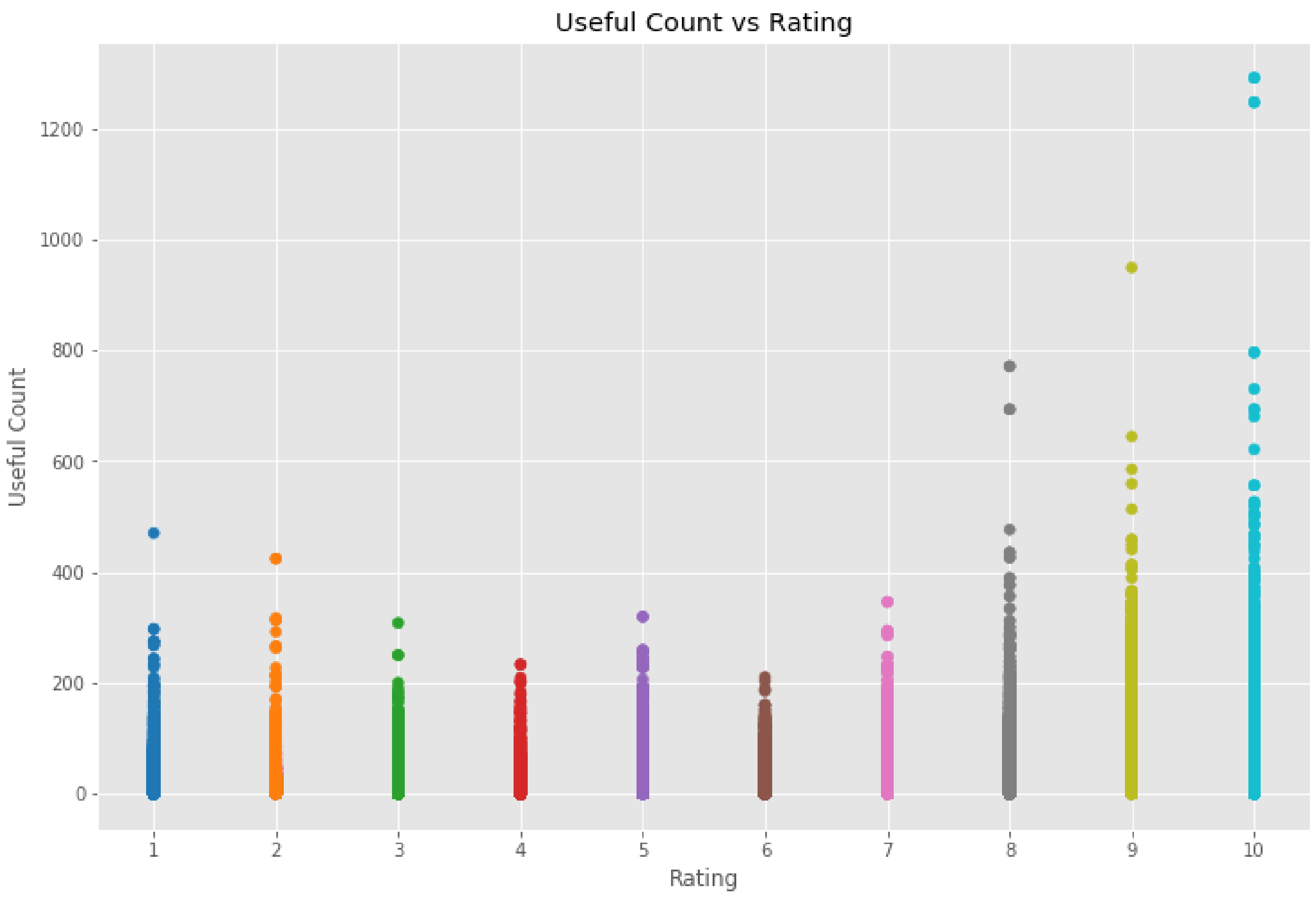
```
Text(0,0.5,'Count')
```

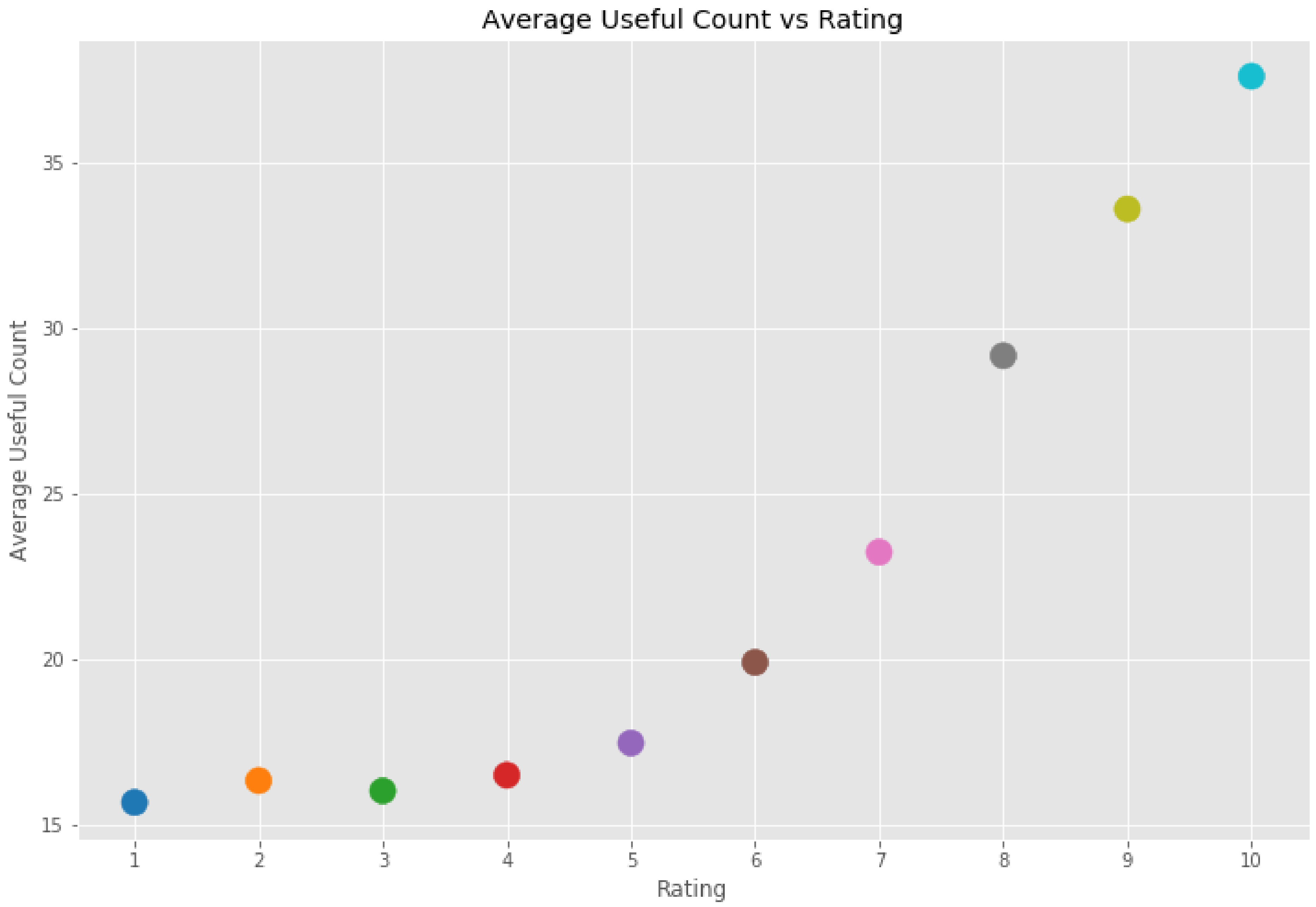


Is rating correlated with the usefulness of the review?

In [16]:

```
# Is rating correlated with usefulness of the review?
plt.scatter(train.rating, train.usefulCount, c=train.rating.values, cmap='tab10')
plt.title('Useful Count vs Rating')
plt.xlabel('Rating')
plt.ylabel('Useful Count')
plt.xticks([i for i in range(1, 11)]);
```




Average Useful Count vs Rating
Average Useful Count
Rating
35
30
25
20
15
1
2
3
4
5
6
7
8
9
10

What makes a review useful? (most useful reviews)

In [19]:

```
# Sort train dataframe from most to least useful
useful_train = train.sort_values(by='usefulCount', ascending=False)
useful_train.iloc[:10]
```

Out[19]:

	uniqueID	drugName	condition	review	rating	date	usefulCount
6716	96616	Sertraline	Depression	"I remember reading people's opinions, on...	10	31-Jul-08	1291
33552	119152	Zoloft	Depression	"I remember reading people's opinions, on...	10	31-Jul-08	1291
21708	131116	Levonorgestrel	Birth Control	"I have had my IUD for over a year now and I t...	10	1-Apr-09	1247
4249	182560	Mirena	Birth Control	"I have had my IUD for over a year now and I t...	10	1-Apr-09	1247
146145	119151	Zoloft	Depression	"I've been on Zoloft 50mg for over two ye...	9	5-Aug-08	949
58608	139141	Phentermine	Weight Loss	"I have used this pill off and on for the past...	10	19-Oct-08	796
16889	52305	Adipex-P	Weight Loss	"I have used this pill off and on for the past...	10	19-Oct-08	796
2039	62757	Citalopram	Depression	"I responded after one week. The side effects ...	8	25-Mar-08	771
152838	89825	Celexa	Depression	"I responded after one week. The side effects ...	8	25-Mar-08	771
5218	107655	Implanon	Birth Control	"I was very nervous about trying Implanon afte...	10	19-Jul-10	730

In [20]:

```python
# Print top 10 most useful reviews
for i in useful_train.review.iloc[:3]:
    print(i, '\n')
```

```
"I remember reading people&#039;s opinions, online, of the drug before I took it and it sca
red me away from it. Then I finally decided to give it a try and it has been the best choic
e I have made. I have been on it for over 4 months and I feel great. I&#039;m on 100mg and
I don&#039;t have any side effects. When I first started I did notice that my hands would t
remble but then it subsided. So honestly, don&#039;t listen to all the negativity because w
hat doesn&#039;t work for some works amazing for others. So go based on youself and not eve
ryone else. It may be a blessing in diguise. The pill is not meant to make you be all happy
go lucky and see &quot;butterflies and roses&quot;, its meant to help put the chemicals in
your mind in balance so you can just be who you are and not overly depressed. I still get s
ad some times, but that is normal, that is life, and it&#039;s up to people to take control
to make a change. I did so by getting on this pill."
```


```
"I remember reading people&#039;s opinions, online, of the drug before I took it and it sca
red me away from it. Then I finally decided to give it a try and it has been the best choic
e I have made. I have been on it for over 4 months and I feel great. I&#039;m on 100mg and
I don&#039;t have any side effects. When I first started I did notice that my hands would t
```

In [21]:

```
# Print 10 of the least useful reviews
for i in useful_train.review.iloc[-3:]:
    print(i, '\n')
```

```
"I started yesterday and today I see it darker. Should I stop? I have a wedding in 10 days... will my melasma be better
by then or still this dark? Thank you"
```

The not-so-useful reviews seem much more negative. The final review listed is barely a review -- just a concerned patient asking questions about the product!

Our conclusions appear consistent with the above graph -- reviewers find higher ratings/better reviews to be more useful than lower ratings/worse reviews. Does this represent some sort of bias within the useful count?

We're also interested in quantifying the sentiment of these reviews.

Sentiment analysis
(= opinion or emotion analysis)

In [22]:
```
sid = SentimentIntensityAnalyzer()
```

In [23]:
```
# Create list (cast to array) of compound polarity sentiment scores for revie
sentiments = []

for i in train.review:
    sentiments.append(sid.polarity_scores(i).get('compound'))

sentiments = np.asarray(sentiments)
```

In [24]:
```
sentiments
```

Out[24]:
```
array([-0.296 ,  0.8603,  0.7645, ..., -0.743 ,  0.6197,  0.6124])
```

[Sentiment analysis - Wikipedia](#)

Natural Language Toolkit ¶

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called "a wonderful tool for teaching, and working in, computational linguistics using Python," and "an amazing library to play with natural language."

Natural Language Processing with Python provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more. The online version of the book has been been updated for Python 3 and NLTK 3. (The original Python 2 version is still available at http://nltk.org/book_1ed.)

https://www.nltk.org/

Add sentiment analysis results to dataset

In [25]:

```
useful_train['sentiment'] = pd.Series(data=sentiments)
```

In [26]:

```
useful_train = useful_train.reset_index(drop=True)
useful_train.head()
```

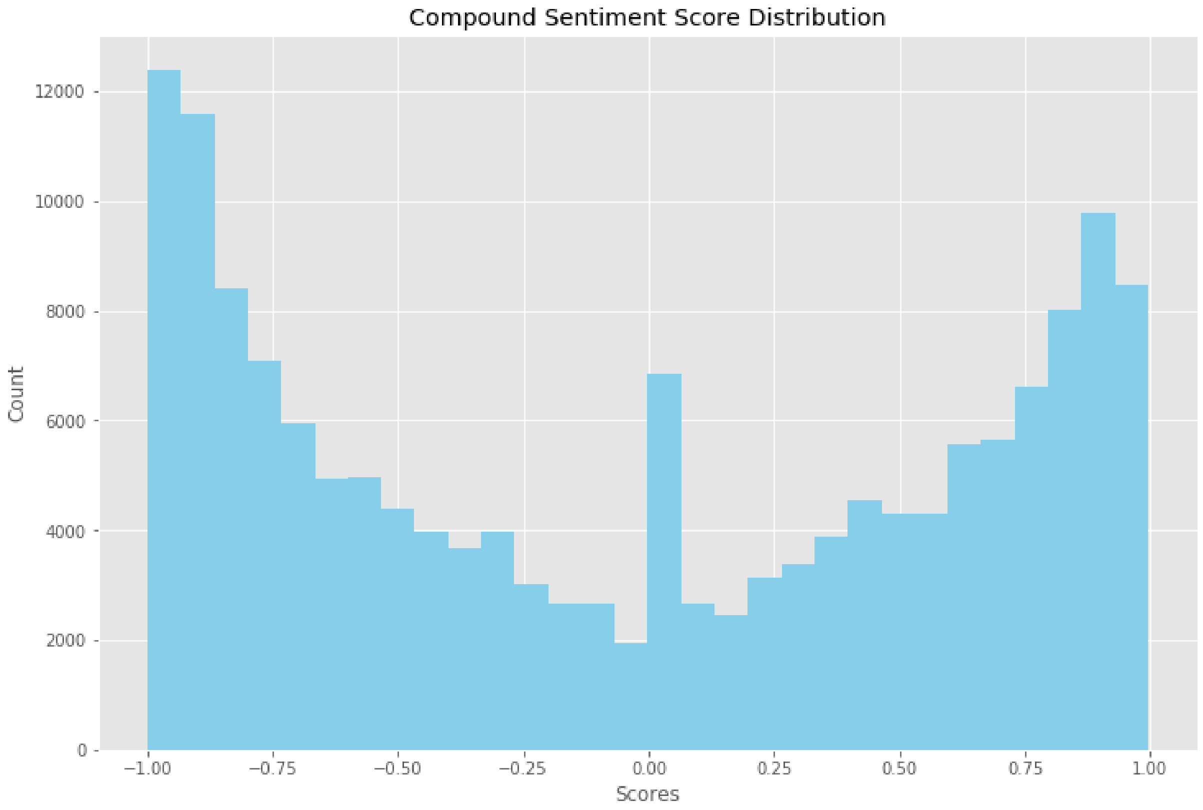
Out[26]:

	uniqueID	drugName	condition	review	rating	date	usefulCount	sentiment
0	96616	Sertraline	Depression	"I remember reading people's opinions, on...	10	31-Jul-08	1291	0.9772
1	119152	Zoloft	Depression	"I remember reading people's opinions, on...	10	31-Jul-08	1291	0.9772
2	131116	Levonorgestrel	Birth Control	"I have had my IUD for over a year now and I t...	10	1-Apr-09	1247	0.7739
3	182560	Mirena	Birth Control	"I have had my IUD for over a year now and I t...	10	1-Apr-09	1247	0.7739
4	119151	Zoloft	Depression	"I've been on Zoloft 50mg for over two ye...	9	5-Aug-08	949	-0.6815

How the sentiment scores are distributed?

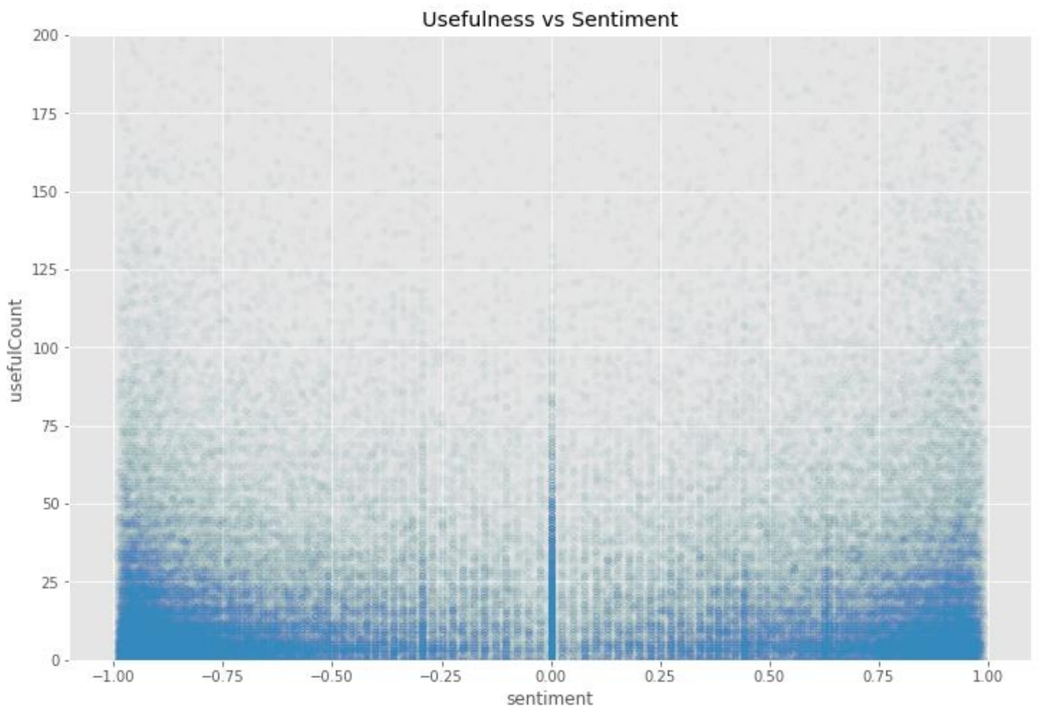
In [27]:

```
useful_train.sentiment.hist(color='skyblue', bins=30)
plt.title('Compound Sentiment Score Distribution')
plt.xlabel('Scores')
plt.ylabel('Count');
```

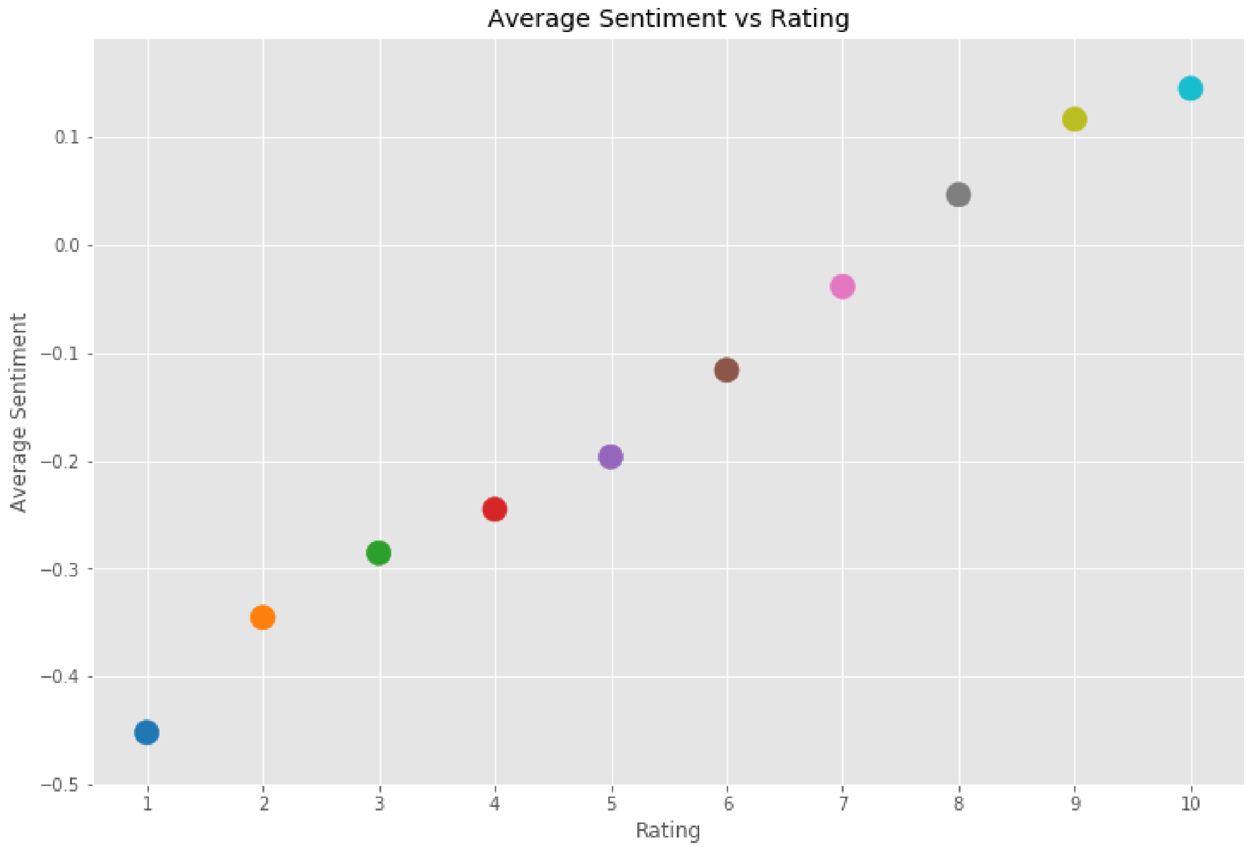


How the sentiment score and usefullness are correlated?

```
In [28]: useful_train.plot(x='sentiment', y='usefulCount', kind='scatter', alpha=0.01)
         plt.title('Usefulness vs Sentiment')
         plt.ylim(0, 200);
```



How does the average sentiment score correlate with rating?



Highest and lowest rated drugs

	0	1
1371	Prevnar 13	3.363636
1372	Fosamax	3.166667
1373	Blisovi 24 Fe	3.088889
1374	Opdivo	3.083333
1375	Miconazole	3.033000
1376	Monistat 7	3.032258
1377	Alendronate	2.954545
1378	Yuvafem	2.318182
1379	Monistat 1-Day or Night Combination Pack	1.416667
1380	ProAir RespiClick	1.193548

	0	1
0	Zutripro	10.000000
1	Chlorpheniramine / hydrocodone / pseudoephedrine	10.000000
2	Silver sulfadiazine	9.972222
3	Drixoral Cold and Allergy	9.948718
4	Dexbrompheniramine / pseudoephedrine	9.947368
5	Emend	9.900000
6	Aprepitant	9.900000
7	Tegaserod	9.812500
8	Zelnorm	9.687500
9	Cyanocobalamin	9.666667

Case 3 – Training problem

Can we predict the rating of the drug based on the review?

Tensorflow classifier example

[Case 3. First classification experiment | Kaggle]()

Tensorflow classifier

- Handling text with tensorflow
- Categorizing the labels
- Split into training and validation sets
- One-hot-coding the labels
- Standard dense NN model
- Training
- Results
- Next steps

Handling text with tensorflow

Text processing

More info:

- scikit-learn CountVectorizer
- scikit-learn text feature extraction
- keras Tokenizer

[6]:
```
# Tokenize the text
samples = train['review']
tokenizer = Tokenizer(num_words = 5000)
tokenizer.fit_on_texts(samples)

# Make one hot samples
data = tokenizer.texts_to_matrix(samples, mode='binary')
```

[7]:
```
# What is the size of the dataset?
data.shape
```

[7]:
```
(15000, 5000)
```

Note: In this demo we use only 15,000 samples from the original dataset!

Categorize labels

Categorize ratings

[8]:
```python
# Create 3 categories
# labels = 2.0, when ratings >= 8
# labels = 1.0, when ratings >= 5 and ratings < 8
# labels = 0.0, when ratings < 5
ratings = train['rating'].values
labels = 1.0*(ratings >= 8) + 1.0*(ratings >= 5)

# Check first 10 of them
labels[:10]
```

[8]:
```
array([2., 0., 2., 1., 2., 2., 2., 2., 2., 2.])
```

Split into training and validation sets

Split to training and validation datasets

```
[9]: train_data, val_data, train_labels, val_labels = train_test_split(data, labels, test_size = 0.250, random_state = 2021)
```

Or you could use validation_split when training the model. Your choice.

One-hot-code the output values

One-hot-code the output values

[10]:
```
# Convert outputs to one-hot-coded categoricals
from tensorflow.keras.utils import to_categorical

train_cat = to_categorical(train_labels)
val_cat = to_categorical(val_labels)

val_cat[:10]
```

[10]:
```
array([[0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.],
       [0., 0., 1.]], dtype=float32)
```

```
# Check first 10 of them
labels[:10]
```

[8]:
```
array([2., 0., 2., 1., 2., 2., 2., 2., 2., 2.])
```

to_categorical converts the numerical labels (0, 1, 2) into one-hot-coded vectors:
0 ➔ [0., 0., 1.]
1 ➔ [0., 1., 0.]
2 ➔ [1., 0., 0.]

Basic dense neural network model

Basic Dense Neural Network (DNN) Model

[11]:
```
# Create a simple sequential model
model = Sequential()
model.add(Dense(256, input_dim = 5000)) # Remember to change the input_dim if you use more words
model.add(Activation('relu'))
model.add(Dense(32)) # Hidden layer
model.add(Activation('relu'))
model.add(Dense(3)) # Output layer has three categories
model.add(Activation('softmax'))

model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['acc'])

model.summary()
```

Notice! We have 3 dense neurons at the botton and 'softmax' activation as we have 3 categories to predict.

Training the model

Training

[12]:
```
%%time
history = model.fit(train_data, train_cat,
                    epochs = 10,
                    batch_size = 128,
                    verbose = 0,
                    validation_data = (val_data, val_cat))
```

```
Wall time: 23.5 s
```

%%time - counts how much time has elapsed during processing the cell.

This demo is using only a subset of the original data to demonstrate the code.

You should use all data in your experiments.

Results

Accuracy and loss trends

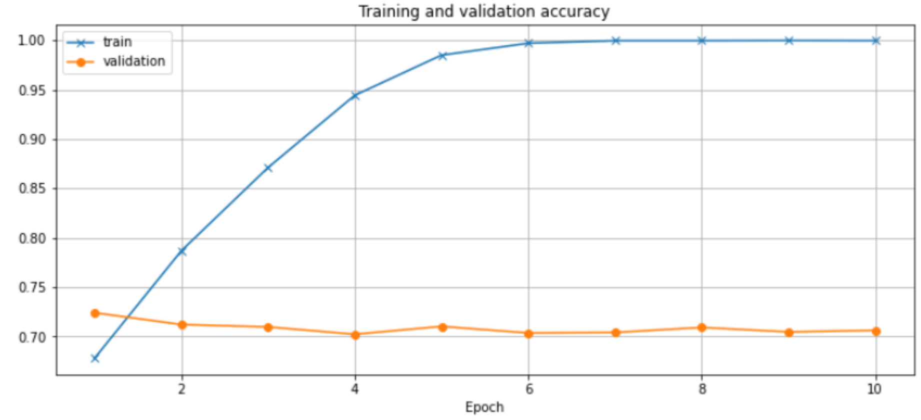
[13]:

```
# Plot the accuracy and loss
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
e = np.arange(len(acc)) + 1

plt.figure()
plt.plot(e, acc, 'x-', label = 'train')
plt.plot(e, val_acc, 'o-', label = 'validation')
plt.title('Training and validation accuracy')
plt.xlabel('Epoch')
plt.grid()
plt.legend()

plt.figure()
plt.plot(e, loss, 'x-', label = 'train')
plt.plot(e, val_loss, 'o-', label = 'validation')
plt.title('Training and validation loss')
plt.xlabel('Epoch')
plt.grid()
plt.legend()

plt.show()
```



Clearly overfits. Starts overfitting from the second epoch. Something needs to be done for this model...

Metrics

Calculate metrics

[14]:
```
# Find the predicted values for the validation set
pred_labels = np.argmax(model.predict(val_data), axis = 1)

# Calculate the classification report
cr = classification_report(val_labels, pred_labels)
print(cr)
```

```
              precision    recall  f1-score   support

         0.0       0.65      0.64      0.64       915
         1.0       0.33      0.23      0.27       556
         2.0       0.79      0.85      0.82      2279

    accuracy                           0.71      3750
   macro avg       0.59      0.57      0.58      3750
weighted avg       0.69      0.71      0.69      3750
```

Support column shows that the categories are uneven. It would help use class weights i

[15]:
```
# Calculate the confusion matrix
cm = confusion_matrix(val_labels, pred_labels).T
print(cm)
```

```
[[ 586  134  185]
 [ 102  127  159]
 [ 227  295 1935]]
```

Lot of space to improve, as can be seen in upper right and lower left corners of the confus

[16]:
```
# Calculate the cohen's kappa, both with linear and quadratic weights
k = cohen_kappa_score(val_labels, pred_labels)
print(f"Cohen's kappa (linear)    = {k:.4f}")
k2 = cohen_kappa_score(val_labels, pred_labels, weights = 'quadratic')
print(f"Cohen's kappa (quadratic) = {k2:.4f}")
```

```
Cohen's kappa (linear)    = 0.4430
Cohen's kappa (quadratic) = 0.5692
```

Summary

In the original article the Kohen's kappa was 83.99% (0.8399). This model is far behind that value. (see Table 2: In-domain Sentiment Analysis). **Please, improve this model!!!**

More info about Kohen's kappa:

- sklearn.metrics.cohen_kappa_score
- Cohen's kappa (Wikipedia)

Next steps

- Use full training dataset
- Try
  - **1D convolutional neural networks (CNN)**
  - recurrent neural networks (RNN)
  - long-short-term-memory networks (LSTM)
- Experiment with number of words in tokenization

- Bonus:
  - Train with one condition, validate with other condition
    - See the Reference article, Table 3 (Cross-domain Sentiment Analysis)

Cohen’s Kappa

Cohen's kappa coefficient (***κ***) is a statistic that is used to measure inter-rater reliability (and also Intra-rater reliability) for qualitative (categorical) items.

It is generally thought to be a more robust measure than simple percent agreement calculation, as *κ* takes into account the possibility of the agreement occurring by chance.

Interpretation

The score lies in the range [-1, +1].

- +1 = complete agreement between the two raters.
- 0 = agreement by chance.
- -1 = complete disagreement between two raters.

Cohen's kappa - Wikipedia

tfa.metrics.CohenKappa | TensorFlow Addons

Example calculation

Suppose that you were analyzing data related to a group of 50 people applying for a grant.

Each grant proposal was **read by two readers** and each reader either said "Yes" or "No" to the proposal.

Suppose the disagreement count data were as follows, where A and B are readers, data on the main diagonal of the matrix (a and d) count the number of agreements and off-diagonal data (b and c) count the number of disagreements:

See Wikipedia, Cohen's Kappa, Simple Example

		B	
		Yes	No
A	Yes	20	5
	No	10	15

		B	
		Yes	No
A	Yes	a	b
	No	c	d

The observed proportionate agreement is:

$$p_o = \frac{a+d}{a+b+c+d} = \frac{20+15}{50} = 0.7$$

To calculate p_e (the probability of random agreement) we note that:

- Reader A said "Yes" to 25 applicants and "No" to 25 applicants. Thus reader A said "Yes" 50% of the time.
- Reader B said "Yes" to 30 applicants and "No" to 20 applicants. Thus reader B said "Yes" 60% of the time.

So the expected probability that both would say yes at random is:

$$p_{\text{Yes}} = \frac{a+b}{a+b+c+d} \cdot \frac{a+c}{a+b+c+d} = 0.5 \times 0.6 = 0.3$$

Similarly:

$$p_{\text{No}} = \frac{c+d}{a+b+c+d} \cdot \frac{b+d}{a+b+c+d} = 0.5 \times 0.4 = 0.2$$

Overall random agreement probability is the probability that they agreed on either Yes or No, i.e.:

$$p_e = p_{\text{Yes}} + p_{\text{No}} = 0.3 + 0.2 = 0.5$$

So now applying our formula for Cohen's Kappa we get:

$$\kappa = \frac{p_o - p_e}{1 - p_e} = \frac{0.7 - 0.5}{1 - 0.5} = 0.4$$

Cohen’s Kappa summary

1. Cohen’s kappa is more informative than overall accuracy when working with unbalanced data.
   - Keep this in mind when you compare or optimize classification models.
2. Cohen’s kappa removes the possibility of the random guess.
   - It measures the number of predictions that cannot be explained by a random guess.
3. The same model will give you lower values of Cohen’s kappa for unbalanced than for balanced test data.
4. Cohen’s kappa says little about the expected accuracy of a single prediction.
   - Cohen’s kappa is not easy to interpret in terms of expected accuracy, and it’s often not recommended to follow any verbal categories as interpretations.

Cohen’s Kappa: What It Is, When to Use It, and How to Avoid Its Pitfalls