In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
df = pd.read_csv('tripadvisor_hotel_reviews.csv')
df.head()
```

Out[2]:

	Review	Rating
0	nice hotel expensive parking got good deal sta...	4
1	ok nothing special charge diamond member hilto...	2
2	nice rooms not 4* experience hotel monaco seat...	3
3	unique, great stay, wonderful time hotel monac...	5
4	great stay great stay, went seahawk game aweso...	5

In [3]:

```
len(df.index)
```

Out[3]:

```
20491
```

In [4]:

```
import numpy as np

def create_sentiment(rating):

    if rating==1 or rating==2:
        return -1 # negative sentiment
    elif rating==4 or rating==5:
        return 1 # positive sentiment
    else:
        return 0 # neutral sentiment

df['Sentiment'] = df['Rating'].apply(create_sentiment)
```

In [ ]:

```
#The target varible that we will be using is "sentiment", this is
#the variable that we will predict the accuracy of
```

In [13]:

```
df.head()
```

Out[13]:

	Review	Rating	Sentiment
0	nice hotel expensive parking got good deal sta...	4	1
1	ok nothing special charge diamond member hilto...	2	-1
2	nice rooms not experience hotel monaco seattl...	3	0
3	unique great stay wonderful time hotel monaco ...	5	1
4	great stay great stay went seahawk game awesom...	5	1

In [ ]:

```
#Function used to remove punctuation, characters and digits
```

In [5]:

```
from sklearn.feature_extraction.text import re

def clean_data(review):

    no_punc = re.sub(r'[^\w\s]', '', review)
    no_digits = ''.join([i for i in no_punc if not i.isdigit()])

    return(no_digits)
```

```
C:\Users\willi\Anaconda3\lib\site-packages\scipy\__init__.py:138: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (det
ected version 1.24.2)
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion} is required for this version of "
```

In [6]:

```
df['Review'][0]
```

Out[6]:

```
'nice hotel expensive parking got good deal stay hotel anniversary, arrived late evening took advice previous reviews did valet parking, check quick easy, lit
tle disappointed non-existent view room room clean nice size, bed comfortable woke stiff neck high pillows, not soundproof like heard music room night morning
loud bangs doors opening closing hear people talking hallway, maybe just noisy neighbors, aveda bath products nice, did not goldfish stay nice touch taken adv
antage staying longer, location great walking distance shopping, overall nice experience having pay 40 parking night,  '
```

In [7]:

```
df['Review'] = df['Review'].apply(clean_data)
df['Review'][0]
```

Out[7]:

```
'nice hotel expensive parking got good deal stay hotel anniversary arrived late evening took advice previous reviews did valet parking check quick easy little
disappointed nonexistent view room room clean nice size bed comfortable woke stiff neck high pillows not soundproof like heard music room night morning loud b
angs doors opening closing hear people talking hallway maybe just noisy neighbors aveda bath products nice did not goldfish stay nice touch taken advantage st
aying longer location great walking distance shopping overall nice experience having pay  parking night  '
```

In [ ]:

```
#TFIDFVectorizer measures how many times a word is repeated across
#a set of documents, the words are eliminated within the entire
#corpus
```

In [8]:

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(strip_accents=None,
                        lowercase=False,
                        preprocessor=None)

X = tfidf.fit_transform(df['Review'])
```

In [ ]:

```
#train-test split
```

In [9]:

```
from sklearn.model_selection import train_test_split
y = df['Sentiment']
X_train, X_test, y_train, y_test = train_test_split(X,y)
```

In [ ]:

```
#Using Logistic Regression
```

In [10]:

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(solver='liblinear')
lr.fit(X_train,y_train)
preds = lr.predict(X_test)
```

In [ ]:

```
#this tells us that the accuracy of our model is ~85%
```

In [11]:

```
from sklearn.metrics import accuracy_score
accuracy_score(preds,y_test)
```

Out[11]:

```
0.8524302166699199
```