In [1]:

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6
qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%
b&scope=email%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.
2Fauth%2Fdrive%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%20https%3A%2F%2Fww
ogleapis.com%2Fauth%2Fpeopleapi.readonly&response_type=code

Enter your authorization code:
··········
Mounted at /content/gdrive
```

In [2]:

```
import pandas as pd
import io


final =  pd.read_csv('gdrive/My Drive/CSV/final_cleaned.csv')
final.head()
```

Out[2]:

	Unnamed: 0	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Sco
0	138706	150524	0006641040	ACITT7DI6IDDL	shari zychinski	0	0	1
1	138683	150501	0006641040	AJ46FKXOVC7NR	Nicholas A Mesiano	2	2	1
2	417839	451856	B00004CXX9	AIUWLEQ1ADEG5	Elizabeth Medina	0	0	1
3	346055	374359	B00004CI84	A344SMIA5JECGM	Vincent P. Ross	1	2	1
4	417838	451855	B00004CXX9	AJH6LUC1UT1ON	The Phantom of the Opera	0	0	1

In [3]:

```
import numpy as np
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers.embeddings import Embedding
from keras.preprocessing import sequence
from keras.layers import Dropout
```

```
Using TensorFlow backend.
```

In [4]:

```
final['Score'].value_counts()
```

Out[4]:

```
 1    307061
-1     57110
Name: Score, dtype: int64
```

In [5]:

```
final.Score.replace(to_replace= -1, value=0, inplace=True)
final['Score'].value_counts()
```

Out[5]:

```
1    307061
0     57110
Name: Score, dtype: int64
```

In [6]:

```
final.head()
```

Out[6]:

	Unnamed: 0	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Sco
0	138706	150524	0006641040	ACITT7DI6IDDL	shari zychinski	0	0	1
1	138683	150501	0006641040	AJ46FKXOVC7NR	Nicholas A Mesiano	2	2	1
2	417839	451856	B00004CXX9	AIUWLEQ1ADEG5	Elizabeth Medina	0	0	1
3	346055	374359	B00004CI84	A344SMIA5JECGM	Vincent P. Ross	1	2	1
4	417838	451855	B00004CXX9	AJH6LUC1UT1ON	The Phantom of the Opera	0	0	1

In [74]:

```
final = final[0:5000] #taking the first 5k reviews
len(final)
```

Out[74]:

```
5000
```

In [75]:

```
z = final['New'].values
z
```

Out[75]:

```
array(['everi book educwitti littl book make son laugh loud recit car drive along alway sing
refrain hes learn whale india droop love new word book introduc silli classic book will bet son st
ill abl recit memori colleg',
       'whole seri great way spend time childrememb see show air televis year ago child sister
later bought day thirti someth use seri book song student teach preschool turn whole school purcha
s along book children tradit live',
       'entertainingl funnibeetlejuic well written movi everyth excel act special effect delight ch
ose view movi',
       ...,
       'must season kitchenfirst tri vegeta hear visit eastern europ late sinc becom kitchen stapl
use vegeta make soup stew chicken veal marsala piccata vegeta use season rice pasta sinc made vege
t use enrich flavor vegetarian thanksgiv stuf vegeta also use enhanc sauc gravi havent tri vegeta
urg believ wide use across europ',
       'year old eat veggiyear old hate lettuc purchas aerogarden watch leav grow alway ask lettuc
longer problem get eat salad love snack leav outsid leav pretti lettuc stay lush full still great
look custom servic also great quick sent replac seed sprout call aerogarden let know pretti outer
lettuc leav week growth immedi sent replac seed kit new seed kit grown full lush much better first
kit told call back problem custom servic realli want complet happi impress make star',
       'super richmmmmmmmm yummi must chocol lover caus chocolatey thick super rich much guess coul
d add milk might well buy cocoa'],
      dtype=object)
```

In [76]:

```
from nltk import FreqDist

#https://stackoverflow.com/questions/41699065/create-vocabulary-dictionary-for-text-mining


train_set = final["New"]
word_dist = FreqDist()
for s in train_set:
    word_dist.update(s.split())

word_dist = dict(word_dist)
word_dist
```

Out[76]:

```
{'everi': 334,
 'book': 91,
 'educwitti': 1,
 'littl': 792,
 'make': 1440,
 'son': 87,
 'laugh': 24,
 'loud': 5,
 'recit': 2,
 'car': 31,
 'drive': 31,
 'along': 91,
 'alway': 323,
 'sing': 15,
 'refrain': 2,
 'hes': 43,
 'learn': 52,
 'whale': 2,
 'india': 25,
 'droop': 3,
 'love': 1291,
 'new': 296,
 'word': 81,
 'introduc': 52,
 'silli': 10,
 'classic': 82,
 'will': 22,
 'bet': 14,
```

```
    'still': 416,
    'abl': 138,
    'memori': 41,
    'colleg': 26,
    'whole': 314,
    'seri': 7,
    'great': 1718,
    'way': 487,
    'spend': 44,
    'time': 926,
    'childrememb': 1,
    'see': 290,
    'show': 81,
    'air': 43,
    'televis': 7,
    'year': 601,
    'ago': 167,
    'child': 72,
    'sister': 22,
    'later': 88,
    'bought': 264,
    'day': 630,
    'thirti': 6,
    'someth': 297,
    'use': 1634,
    'song': 14,
    'student': 25,
    'teach': 12,
    'preschool': 2,
    'turn': 156,
    'school': 53,
    'purchas': 257,
    'children': 80,
    'tradit': 133,
    'live': 211,
    'entertainingl': 1,
    'funnibeetlejuic': 1,
    'well': 721,
    'written': 16,
    'movi': 161,
    'everyth': 125,
    'excel': 405,
    'act': 27,
    'special': 152,
    'effect': 110,
    'delight': 89,
    'chose': 6,
    'view': 14,
    'modern': 16,
    'fairi': 1,
    'taletwist': 1,
    'rumplestiskin': 1,
    'captur': 6,
    'film': 117,
    'star': 226,
    'michael': 30,
    'keaton': 39,
    'geena': 22,
    'davi': 26,
    'prime': 13,
    'tim': 47,
    'burton': 67,
    'masterpiec': 9,
    'rumbl': 1,
    'absurd': 6,
    'wonder': 488,
    'pace': 8,
    'point': 112,
    'dull': 7,
    'moment': 34,
    'fantastbeetlejuic': 1,
    'funni': 46,
    'hilari': 11,
    'wacki': 4,
    'beetlejuic': 64,
    'help': 267,
    'think': 515,
```

```
'one': 1738,
'best': 1192,
'ever': 484,
'made': 529,
'sure': 222,
'youll': 208,
'agre': 53,
'good': 1871,
'watch': 103,
'greatone': 2,
'collect': 30,
'fill': 197,
'comedi': 22,
'action': 6,
'whatev': 58,
'els': 138,
'want': 520,
'call': 184,
'clamshel': 3,
'edit': 19,
'versionalway': 1,
'enjoy': 480,
'entertain': 25,
'didnt': 237,
'hesit': 23,
'pick': 113,
'guess': 72,
'market': 188,
'plan': 71,
'famili': 242,
'elimin': 18,
'strong': 224,
'profan': 1,
'element': 18,
'usual': 195,
'version': 161,
'warn': 60,
'uncut': 7,
'avoid': 62,
'death': 28,
'flibought': 1,
'apart': 37,
'infest': 2,
'fruit': 235,
'fli': 16,
'hour': 127,
'trap': 38,
'mani': 477,
'within': 62,
'practic': 42,
'gone': 58,
'may': 277,
'long': 305,
'term': 26,
'solut': 9,
'crazi': 49,
'consid': 109,
'buy': 799,
'surfac': 30,
'sticki': 24,
'tri': 1511,
'touch': 64,
'bettlejuic': 5,
'bettlejuichappen': 1,
'say': 364,
'name': 190,
'three': 153,
'keaten': 1,
'two': 416,
'coupl': 139,
'old': 235,
'stori': 47,
'hous': 169,
'come': 464,
'back': 326,
'suppli': 44,
'store': 551,
```

'sudden': 21,
'get': 1243,
'caught': 27,
'insid': 92,
'bridg': 5,
'start': 242,
'tumbl': 1,
'lake': 8,
'board': 9,
'catch': 11,
'theyv': 23,
'got': 284,
'hope': 131,
'small': 274,
'dog': 421,
'step': 22,
'slide': 10,
'water': 513,
'minut': 257,
'find': 784,
'home': 233,
'somehow': 18,
'somehad': 1,
'light': 250,
'fireplac': 3,
'done': 65,
'magic': 44,
'dead': 29,
'guy': 58,
'known': 61,
'appear': 49,
'survic': 1,
'soon': 74,
'wish': 108,
'never': 429,
'troublemak': 2,
'save': 136,
'account': 8,
'said': 145,
'cant': 394,
'leav': 243,
'theirselv': 1,
'anoth': 245,
'world': 173,
'giant': 16,
'sandworm': 3,
'stellar': 3,
'awesom': 109,
'play': 74,
'lead': 26,
'role': 13,
'researchget': 1,
'realli': 874,
'imposs': 26,
'today': 82,
'french': 70,
'vhs': 5,
'could': 396,
'pleas': 172,
'tell': 127,
'tks': 2,
'research': 36,
'beatlejuic': 3,
'video': 10,
'versionget': 1,
'look': 528,
'productrealli': 6,
'idea': 79,
'final': 126,
'product': 1395,
'outstand': 61,
'decal': 2,
'window': 16,
'everybodi': 19,
'ask': 116,
'thumb': 14,
'wow': 65,

```
'slickerreceiv': 1,
'shipment': 35,
'hard': 303,
'wait': 129,
'instead': 171,
'sticker': 7,
'remov': 64,
'easili': 97,
'daughter': 91,
'design': 43,
'sign': 13,
'print': 14,
'revers': 6,
'beauti': 116,
'shop': 95,
'program': 16,
'go': 206,
'lot': 355,
'fun': 128,
'everywher': 28,
'like': 2392,
'screen': 17,
'comput': 12,
'monitor': 40,
'end': 144,
'gopher': 12,
'problemrecent': 1,
'woodstream': 2,
'corp': 1,
'lay': 6,
'easi': 309,
'set': 84,
'work': 432,
'success': 48,
'also': 964,
'rememb': 100,
'wire': 2,
'attach': 14,
'tie': 6,
'steak': 47,
'prevent': 36,
'drag': 7,
'hole': 19,
'luck': 18,
'musteasi': 1,
'mess': 46,
'offer': 152,
'vibrant': 6,
'color': 152,
'taint': 4,
'decort': 1,
'would': 858,
'high': 554,
'recommend': 545,
'anyon': 156,
'decor': 19,
'usemuch': 1,
'easier': 78,
'wilson': 1,
'past': 126,
'frost': 52,
'simpl': 119,
'complaint': 35,
'must': 185,
'often': 108,
'fresh': 396,
'origin': 170,
'master': 13,
'storytel': 3,
'burtongreat': 1,
'dont': 806,
'even': 800,
'know': 447,
'sum': 7,
'first': 545,
'complet': 108,
'unlik': 78,
```

```
'seen': 74,
'second': 158,
'kind': 220,
'spooki': 1,
'weird': 29,
'feel': 257,
'probabl': 205,
'art': 39,
'director': 17,
'welch': 1,
'anyth': 211,
'invent': 22,
'fabul': 45,
'comed': 4,
'fanasi': 1,
'direct': 111,
'masterbeetlejuic': 1,
'amus': 9,
'romp': 1,
'explor': 4,
'incred': 69,
'possibl': 77,
'boundari': 3,
'tale': 11,
'recent': 105,
'marri': 10,
'led': 6,
'chaotic': 3,
'supernatur': 5,
'adam': 15,
'barbara': 11,
'maitland': 18,
'alec': 28,
'baldwin': 30,
'discov': 81,
'conflict': 2,
'rather': 111,
'human': 31,
'imperfect': 2,
'haunt': 7,
'plagu': 3,
'afterlif': 11,
'project': 6,
'seem': 282,
'blind': 6,
'assign': 2,
'dispassion': 1,
'filmmak': 1,
'commerci': 35,
'reason': 173,
'plot': 9,
'bizarr': 12,
'subject': 4,
'matter': 57,
'remark': 10,
'complement': 20,
'unusu': 30,
'macabr': 2,
'artist': 3,
'sensibl': 5,
'extraordinarili': 2,
'creat': 77,
'unbeliev': 20,
'brilliant': 11,
'guidanc': 1,
'imagin': 64,
'pee': 14,
'wee': 3,
'big': 230,
'adventur': 10,
'batman': 14,
'wood': 21,
'sleepi': 8,
'hollow': 4,
'uniqu': 92,
'creativ': 16,
'landscap': 2,
```

'culmin': 1,
'essenti': 40,
'abund': 7,
'ironi': 4,
'outlandish': 1,
'yet': 200,
'behavior': 2,
'grace': 10,
'bodi': 99,
'augment': 3,
'devious': 1,
'energet': 6,
'perform': 33,
'glenn': 3,
'shadix': 4,
'jeffrey': 9,
'jone': 14,
'winona': 21,
'ryder': 22,
'catherin': 13,
'hara': 2,
'bustl': 1,
'uninhibit': 1,
'hilar': 2,
'persist': 4,
'push': 10,
'level': 84,
'almost': 212,
'affabl': 1,
'euphoria': 1,
'pair': 15,
'ingeni': 3,
'screenplay': 1,
'tour': 3,
'forc': 17,
'transform': 9,
'exuber': 1,
'jovial': 1,
'exercis': 22,
'extrem': 87,
'satisfi': 160,
'philosoph': 1,
'percept': 1,
'though': 286,
'unabl': 17,
'undeserv': 1,
'deep': 55,
'critic': 15,
'analysi': 3,
'undeni': 1,
'inspir': 24,
'concept': 21,
'flawless': 3,
'transfer': 14,
'dvd': 36,
'packag': 526,
'includ': 180,
'theatric': 5,
'trailer': 4,
'isol': 10,
'danni': 8,
'elfman': 8,
'music': 13,
'track': 26,
'choic': 136,
'anamorph': 1,
'widescreen': 5,
'pan': 50,
'scan': 2,
'eventu': 15,
'need': 405,
'wiltonsick': 1,
'scad': 1,
'nasti': 41,
'toothpick': 27,
'counter': 16,
'tint': 1,

'overdu': 1,
'except': 87,
'welcom': 13,
'offens': 7,
'tast': 2442,
'blend': 349,
'opinion': 60,
'smooth': 162,
'wilton': 1,
'experi': 172,
'bit': 373,
'hue': 5,
'right': 321,
'worth': 251,
'funnithought': 1,
'geeki': 1,
'husband': 106,
'priceless': 1,
'scenc': 1,
'kid': 237,
'successmani': 1,
'dealt': 4,
'figur': 58,
'angel': 9,
'clarenc': 1,
'life': 175,
'brad': 2,
'pitt': 1,
'abysm': 1,
'joe': 26,
'black': 327,
'howev': 282,
'present': 49,
'interestng': 1,
'town': 21,
'die': 41,
'oddest': 1,
'return': 60,
'thier': 12,
'selv': 1,
'confin': 3,
'premis': 4,
'send': 59,
'strang': 48,
'desert': 21,
'popul': 3,
'remain': 50,
'wouldnt': 85,
'bad': 236,
'wasnt': 89,
'nyc': 11,
'artisan': 3,
'move': 67,
'remodel': 1,
'place': 177,
'consult': 4,
'juno': 3,
'sylvia': 3,
'sidney': 4,
'hire': 6,
'betelgues': 2,
'pronouc': 1,
'henc': 9,
'titl': 14,
'unfortun': 57,
'po': 1,
'demon': 4,
'refus': 19,
'underworld': 1,
'visual': 10,
'stun': 4,
'micheal': 3,
'give': 471,
'blast': 12,
'beer': 72,
'swill': 1,
'cuss': 1,

'worst': 44,
'nightmar': 11,
'superpow': 1,
'proud': 21,
'martha': 2,
'stewart': 7,
'style': 71,
'cake': 203,
'cookidont': 1,
'liquid': 48,
'food': 924,
'intens': 48,
'shade': 10,
'ice': 223,
'season': 175,
'cooki': 341,
'excit': 36,
'tabl': 46,
'dessert': 66,
'parti': 66,
'tea': 2662,
'event': 10,
'especi': 192,
'contest': 4,
'birthday': 35,
'christma': 90,
'raini': 11,
'afternoon': 60,
'casper': 1,
'ghostmichael': 1,
'bring': 105,
'distinguish': 8,
'characterist': 11,
'ghoul': 1,
'mere': 12,
'script': 4,
'cinematographi': 1,
'focus': 26,
'snack': 415,
'popcorn': 64,
'terribl': 43,
'dvdcontinu': 1,
'amaz': 168,
'shoddi': 2,
'treatment': 11,
'releas': 35,
'simpli': 117,
'disgrac': 1,
'exampl': 25,
'energi': 110,
'extra': 183,
'mention': 66,
'delet': 3,
'scene': 18,
'featurett': 1,
'lousi': 6,
'commentari': 7,
'wors': 31,
'cut': 112,
'less': 318,
'theater': 13,
'advic': 15,
'money': 117,
'somebodi': 13,
'capabl': 5,
'medium': 65,
'provid': 91,
'materi': 24,
'greatmovi': 1,
'cheat': 4,
'dvdwarn': 1,
'trick': 23,
'format': 7,
'compani': 277,
'mistak': 25,
'full': 189,
'compar': 146,

```
 'compul': 110,
 'comtain': 1,
 'pictur': 48,
 'top': 214,
 'bottom': 59,
 'mean': 126,
 'take': 393,
 'slight': 151,
 'expect': 153,
 'care': 150,
 'rip': 11,
 'peopl': 292,
 'nice': 433,
 'bright': 21,
 'colorbought': 1,
 'dia': 1,
 'los': 5,
 'muerto': 1,
 'skull': 3,
 'ateco': 2,
 'gave': 102,
 'mix': 586,
 'total': 131,
 'beatlejuicwinona': 1,
 'hot': 592,
 'gothic': 4,
 'princess': 3,
 'doom': 3,
 'eldest': 1,
 'brother': 26,
 'ive': 608,
 'zenith': 1,
 'goe': 114,
 'miss': 77,
 'cours': 131,
 'fan': 142,
 'fav': 6,
 'part': 154,
 'chum': 1,
 'charact': 31,
 'wear': 15,
 'type': 154,
 'cloth': 11,
 'deal': 92,
 'witti': 3,
 'clever': 9,
 'charm': 16,
 'spit': 28,
 'hit': 69,
 'gina': 3,
 'favoritfavorit': 1,
 'mine': 98,
 'girl': 36,
 'read': 150,
 'mom': 69,
 'bedtim': 11,
 'copi': 12,
 'either': 132,
 'lost': 58,
 'given': 75,
 'away': 189,
 'found': 506,
 'garag': 1,
 'sale': 49,
 'condit': 63,
 'around': 216,
 'somewher': 14,
 'attic': 1,
 'niec': 3,
 'piec': 155,
 'heavenyear': 1,
 'simul': 5,
 'truli': 106,
 'italian': 76,
 'espresso': 144,
 'cappuccino': 9,
 'without': 390,
 'sever': 207,
```

'arriv': 164,
'sent': 60,
'briel': 3,
'cadiz': 1,
'machin': 207,
'russian': 7,
'china': 46,
'cup': 585,
'case': 192,
'espression': 11,
'pod': 477,
'kindest': 1,
'qualiti': 316,
'eas': 26,
'exquisit': 18,
'prepar': 113,
'gift': 239,
'itali': 32,
'heaven': 59,
'earth': 18,
'everyonvacat': 1,
'meet': 24,
'demis': 1,
'kill': 24,
'local': 265,
'worri': 60,
'head': 64,
'transport': 8,
'mysteri': 22,
'journey': 7,
'roller': 2,
'coaster': 1,
'score': 14,
'til': 5,
'cri': 9,
'odd': 34,
'fantasi': 6,
'comeditwo': 1,
'accid': 16,
'aliv': 10,
'ghost': 18,
'stuck': 43,
'etern': 2,
'month': 246,
'york': 30,
'jefferi': 2,
'connecticut': 2,
'farmhous': 1,
'chang': 127,
'aesthet': 5,
'control': 50,
'terrif': 45,
'box': 698,
'offic': 53,
'summer': 54,
'plenti': 39,
'humor': 14,
'win': 12,
'oscar': 9,
'likeabl': 3,
'age': 56,
'gross': 20,
'funmichael': 1,
'alreadi': 75,
'major': 35,
'solidifi': 1,
'status': 8,
'hollywood': 7,
'brightest': 2,
'generat': 15,
'belli': 27,
'wife': 60,
'surpris': 152,
'centerpiec': 1,
'gag': 13,
'defin': 39,
'begin': 40

```
'begin': 40,
'ad': 306,
'hilarihilari': 1,
'favorit': 504,
'dine': 7,
'room': 48,
'danc': 15,
'rider': 2,
'spectacular': 12,
'couldnt': 93,
'stop': 119,
'keep': 381,
'stitch': 2,
'night': 92,
'cartoon': 7,
'shown': 21,
'decid': 131,
'fact': 154,
'hey': 19,
'much': 717,
'thing': 466,
'wrong': 110,
'short': 45,
'mayb': 122,
'efect': 1,
'older': 23,
'middl': 32,
'nameset': 1,
'england': 21,
'follow': 73,
'maitlin': 1,
'owner': 28,
'larg': 130,
'hardwar': 2,
'vacat': 10,
'drown': 8,
'mishap': 1,
'taken': 39,
'boister': 1,
'deet': 1,
'prospect': 3,
'poltergeist': 1,
'featur': 26,
'cast': 18,
'young': 39,
'storylin': 3,
'boast': 9,
'believ': 128,
'prop': 2,
'costum': 1,
'absolut': 169,
'liter': 38,
'diebeetlejuic': 1,
'funiest': 1,
'realiti': 9,
'romanc': 1,
'deceas': 6,
'handbook': 5,
'break': 54,
'cover': 71,
'job': 50,
'monthbook': 1,
'poetri': 5,
'cute': 28,
'poem': 5,
'author': 8,
'purpos': 25,
'write': 61,
'rhythm': 3,
'thingwine': 1,
'saver': 16,
'obvious': 42,
'open': 203,
'bottl': 237,
'wine': 89,
'stay': 115,
'glass': 71,
'dinner': 93
```

```
'dinner': 93,
'let': 180,
'expens': 153,
'sinc': 487,
'wont': 213,
'wast': 66,
'rest': 60,
'interest': 108,
'undrunk': 1,
'portion': 42,
'next': 119,
'put': 378,
'fridg': 39,
'drink': 605,
'pull': 30,
'red': 221,
'guynever': 1,
'dissapoint': 20,
'gadget': 11,
'concerto': 3,
'white': 226,
'stopper': 6,
'handi': 29,
'click': 16,
'sound': 61,
'vacuum': 10,
'correct': 32,
'neat': 10,
'beetlejuicsay': 1,
'actual': 269,
'memor': 8,
'drama': 4,
'pervert': 1,
'caus': 74,
'quit': 279,
'chao': 3,
'newli': 2,
'couldv': 3,
'better': 688,
'disapoint': 6,
'beetljuic': 1,
'footag': 2,
'havent': 83,
'lover': 88,
'starbuck': 96,
'poduse': 2,
'chamonix': 1,
'amazon': 549,
'produc': 94,
'delici': 513,
'latt': 26,
'smell': 234,
'regular': 280,
'decaf': 63,
'dark': 246,
'rich': 217,
'roast': 224,
'fyi': 3,
'paper': 84,
'tab': 10,
'burn': 55,
'finger': 18,
'individu': 121,
'wrap': 92,
'space': 14,
'your': 193,
'stamp': 7,
'date': 80,
'ghostsimpli': 1,
'funniest': 6,
'ghostbust': 1,
'mischief': 1,
'rid': 19,
'decemb': 8,
'snowman': 2,
'anniversaridaughter': 1,
'rosi': 2,
```

```
 'carol': 2,
 'king': 24,
 'avail': 209,
 'far': 270,
 'johnni': 4,
 'allig': 1,
 'chicken': 264,
 'soup': 165,
 'rice': 241,
 'mauric': 4,
 'sendak': 10,
 'plus': 117,
 'cheap': 60,
 'movimovi': 1,
 'doesnt': 261,
 'lydia': 13,
 'deitz': 1,
 'sad': 31,
 'depress': 7,
 'mother': 54,
 'redecor': 1,
 'ohara': 10,
 'other': 96,
 'march': 10,
 'blooper': 1,
 'workskeptic': 1,
 'item': 224,
 'prior': 12,
 'vacuvin': 6,
 'last': 270,
 'maximum': 7,
 'immedi': 61,
 'longer': 119,
 'refriger': 56,
 'week': 202,
 'devic': 3,
 'pump': 35,
 'plastic': 55,
 'cork': 3,
 'pressur': 29,
 'rins': 37,
 'perfect': 430,
 'singl': 114,
 'sit': 96,
 'switch': 60,
 'betterive': 2,
 'vacu': 11,
 'vin': 11,
 'previous': 51,
 ...}
```

In [77]:

```
type(word_dist)
```

Out[77]:

```
dict
```

In [78]:

```
import operator
word_dist = sorted(word_dist.items(), key=operator.itemgetter(1),reverse = True)
word_dist
```

Out[78]:

```
[('tea', 2662),
 ('tast', 2442),
 ('like', 2392),
 ('flavor', 1996),
 ('good', 1871),
 ('one', 1738),
 ('great', 1718),
 ('use', 1634),
```

```
    ('tri', 1511),
    ('make', 1440),
    ('product', 1395),
    ('love', 1291),
    ('coffe', 1247),
    ('get', 1243),
    ('best', 1192),
    ('eat', 1024),
    ('also', 964),
    ('time', 926),
    ('food', 924),
    ('chocol', 913),
    ('realli', 874),
    ('would', 858),
    ('dont', 806),
    ('even', 800),
    ('buy', 799),
    ('littl', 792),
    ('find', 784),
    ('well', 721),
    ('much', 717),
    ('box', 698),
    ('better', 688),
    ('bag', 639),
    ('sweet', 635),
    ('day', 630),
    ('order', 620),
    ('sugar', 615),
    ('ive', 608),
    ('drink', 605),
    ('year', 601),
    ('cat', 594),
    ('hot', 592),
    ('mix', 586),
    ('cup', 585),
    ('sauc', 557),
    ('high', 554),
    ('store', 551),
    ('bar', 550),
    ('amazon', 549),
    ('recommend', 545),
    ('first', 545),
    ('made', 529),
    ('look', 528),
    ('packag', 526),
    ('want', 520),
    ('chees', 517),
    ('think', 515),
    ('water', 513),
    ('delici', 513),
    ('found', 506),
    ('favorit', 504),
    ('price', 502),
    ('wonder', 488),
    ('way', 487),
    ('sinc', 487),
    ('ever', 484),
    ('enjoy', 480),
    ('mani', 477),
    ('pod', 477),
    ('candi', 474),
    ('give', 471),
    ('milk', 467),
    ('thing', 466),
    ('come', 464),
    ('brand', 460),
    ('natur', 457),
    ('differ', 453),
    ('know', 447),
    ('treat', 436),
    ('nice', 433),
    ('work', 432),
    ('add', 431),
    ('perfect', 430),
    ('never', 429),
    ('dog', 421),
    ('still', 416),
```

```
 ('two', 416),
 ('snack', 415),
 ('excel', 405),
 ('need', 405),
 ('green', 405),
 ('could', 396),
 ('fresh', 396),
 ('cant', 394),
 ('take', 393),
 ('without', 390),
 ('organ', 385),
 ('stuff', 384),
 ('keep', 381),
 ('review', 381),
 ('put', 378),
 ('ingredi', 376),
 ('bit', 373),
 ('say', 364),
 ('lot', 355),
 ('blend', 349),
 ('healthi', 349),
 ('ship', 349),
 ('cooki', 341),
 ('contain', 337),
 ('everi', 334),
 ('black', 327),
 ('back', 326),
 ('alway', 323),
 ('tasti', 323),
 ('serv', 323),
 ('right', 321),
 ('less', 318),
 ('qualiti', 316),
 ('whole', 314),
 ('enough', 312),
 ('oil', 311),
 ('fat', 311),
 ('easi', 309),
 ('real', 307),
 ('ad', 306),
 ('long', 305),
 ('hard', 303),
 ('diet', 302),
 ('someth', 297),
 ('new', 296),
 ('peopl', 292),
 ('cook', 292),
 ('see', 290),
 ('calori', 289),
 ('though', 286),
 ('got', 284),
 ('varieti', 284),
 ('butter', 284),
 ('seem', 282),
 ('howev', 282),
 ('regular', 280),
 ('quit', 279),
 ('may', 277),
 ('compani', 277),
 ('free', 276),
 ('noodl', 275),
 ('small', 274),
 ('pack', 273),
 ('far', 270),
 ('last', 270),
 ('actual', 269),
 ('help', 267),
 ('cereal', 267),
 ('local', 265),
 ('bought', 264),
 ('chicken', 264),
 ('quick', 262),
 ('doesnt', 261),
 ('purchas', 257),
 ('minut', 257),
 ('feel', 257),
 ('textur', 257),
```

```
('friend', 255),
('juic', 253),
('bread', 253),
('worth', 251),
('light', 250),
('dri', 250),
('senseo', 250),
('month', 246),
('dark', 246),
('anoth', 245),
('pill', 245),
('leav', 243),
('famili', 242),
('start', 242),
('salt', 242),
('rice', 241),
('gift', 239),
('theyr', 239),
('didnt', 237),
('kid', 237),
('bottl', 237),
('groceri', 237),
('peanut', 237),
('bad', 236),
('fruit', 235),
('old', 235),
('smell', 234),
('home', 233),
('chew', 231),
('big', 230),
('orang', 228),
('cream', 228),
('star', 226),
('white', 226),
('strong', 224),
('roast', 224),
('item', 224),
('ice', 223),
('thank', 223),
('health', 223),
('vanilla', 223),
('pretti', 223),
('sure', 222),
('red', 221),
('kind', 220),
('brew', 218),
('breakfast', 218),
('rich', 217),
('around', 216),
('low', 216),
('amount', 215),
('definit', 215),
('top', 214),
('problem', 214),
('wont', 213),
('almost', 212),
('live', 211),
('anyth', 211),
('protein', 211),
('avail', 209),
('pasta', 209),
('youll', 208),
('sever', 207),
('machin', 207),
('go', 206),
('probabl', 205),
('cake', 203),
('open', 203),
('spice', 203),
('week', 202),
('size', 202),
('bean', 202),
('per', 201),
('yet', 200),
('gum', 200),
('morn', 200),
('fill', 197),
```

```
('usual', 195),
('recip', 195),
('your', 193),
('especi', 192),
('case', 192),
('lemon', 192),
('that', 191),
('wheat', 191),
('name', 190),
('full', 189),
('away', 189),
('market', 188),
('must', 185),
('hand', 185),
('call', 184),
('extra', 183),
('happi', 183),
('meal', 182),
('powder', 182),
('includ', 180),
('let', 180),
('noth', 180),
('can', 179),
('place', 177),
('life', 175),
('season', 175),
('world', 173),
('reason', 173),
('might', 173),
('mouth', 173),
('pleas', 172),
('experi', 172),
('instead', 171),
('origin', 170),
('hous', 169),
('absolut', 169),
('prefer', 169),
('came', 169),
('amaz', 168),
('licoric', 168),
('ago', 167),
('cracker', 167),
('thought', 167),
('soup', 165),
('arriv', 164),
('fine', 164),
('smooth', 162),
('soft', 162),
('movi', 161),
('version', 161),
('satisfi', 160),
('person', 160),
('bitter', 159),
('spici', 159),
('soda', 159),
('second', 158),
('half', 158),
('turn', 156),
('anyon', 156),
('piec', 155),
('part', 154),
('type', 154),
('fact', 154),
('receiv', 154),
('honey', 154),
('three', 153),
('expect', 153),
('expens', 153),
('special', 152),
('offer', 152),
('color', 152),
('surpris', 152),
('salad', 152),
('bake', 152),
('slight', 151),
('tazo', 151),
('nut', 151),
```

```
 ('care', 150),
 ('read', 150),
 ('weight', 150),
 ('syrup', 150),
 ('nutrit', 150),
 ('sweeten', 148),
 ('compar', 146),
 ('said', 145),
 ('end', 144),
 ('espresso', 144),
 ('oliv', 143),
 ('fan', 142),
 ('addict', 142),
 ('isnt', 142),
 ('base', 141),
 ('chai', 140),
 ('coupl', 139),
 ('cinnamon', 139),
 ('abl', 138),
 ('els', 138),
 ('least', 138),
 ('heat', 138),
 ('corn', 138),
 ('crunchi', 137),
 ('save', 136),
 ('choic', 136),
 ('meat', 136),
 ('carri', 134),
 ('cost', 134),
 ('tradit', 133),
 ('disappoint', 133),
 ('altern', 133),
 ('either', 132),
 ('hope', 131),
 ('total', 131),
 ('cours', 131),
 ('decid', 131),
 ('addit', 131),
 ('dish', 131),
 ('larg', 130),
 ('bowl', 130),
 ('cocoa', 130),
 ('chip', 130),
 ('wait', 129),
 ('shape', 129),
 ('fun', 128),
 ('believ', 128),
 ('mild', 128),
 ('particular', 128),
 ('cold', 128),
 ('hour', 127),
 ('tell', 127),
 ('chang', 127),
 ('fast', 127),
 ('yummi', 127),
 ('final', 126),
 ('past', 126),
 ('mean', 126),
 ('run', 126),
 ('everyth', 125),
 ('clean', 125),
 ('tree', 125),
 ('sell', 124),
 ('tomato', 124),
 ('egg', 123),
 ('mayb', 122),
 ('bite', 122),
 ('pop', 122),
 ('becom', 122),
 ('coke', 122),
 ('individu', 121),
 ('although', 121),
 ('everyon', 121),
 ('side', 120),
 ('jerki', 120),
 ('numi', 120),
 ('simpl', 119),
```

```
('stop', 119),
('next', 119),
('longer', 119),
('soy', 119),
('root', 119),
('exact', 119),
('caffein', 119),
('aroma', 118),
('ground', 118),
('film', 117),
('simpli', 117),
('money', 117),
('plus', 117),
('stock', 117),
('ask', 116),
('beauti', 116),
('ill', 116),
('beef', 116),
('carb', 116),
('process', 116),
('stay', 115),
('jar', 115),
('goe', 114),
('singl', 114),
('pound', 114),
('pick', 113),
('prepar', 113),
('ginger', 113),
('gluten', 113),
('pocket', 113),
('point', 112),
('cut', 112),
('went', 112),
('direct', 111),
('rather', 111),
('brown', 111),
('packet', 111),
('english', 111),
('fiber', 111),
('effect', 110),
('energi', 110),
('wrong', 110),
('grey', 110),
('sometim', 110),
('consid', 109),
('awesom', 109),
('near', 109),
('consist', 109),
('suggest', 109),
('note', 109),
('wish', 108),
('often', 108),
('complet', 108),
('interest', 108),
('rate', 108),
('flower', 108),
('vitamin', 107),
('com', 107),
('list', 107),
('husband', 106),
('truli', 106),
('notic', 106),
('artifici', 106),
('normal', 106),
('cherri', 106),
('roll', 106),
('herbal', 106),
('recent', 105),
('bring', 105),
('plain', 105),
('valu', 105),
('flour', 105),
('true', 104),
('gummi', 104),
('watch', 103),
('herb', 103),
('creami', 103),
```

```
('gave', 102),
('close', 102),
('five', 102),
('super', 102),
('pepper', 102),
('general', 101),
('state', 101),
('combin', 101),
('custom', 101),
('servic', 101),
('rememb', 100),
('pleasant', 100),
('pot', 100),
('seed', 100),
('import', 100),
('acid', 100),
('bodi', 99),
('fantast', 99),
('mine', 98),
('etc', 98),
('maker', 98),
('easili', 97),
('balanc', 97),
('pay', 97),
('plant', 97),
('dress', 97),
('starbuck', 96),
('other', 96),
('sit', 96),
('glad', 96),
('microwav', 96),
('sodium', 96),
('shop', 95),
('certain', 95),
('pure', 95),
('stick', 95),
('substitut', 95),
('entir', 95),
('four', 95),
('cool', 95),
('earl', 95),
('produc', 94),
('instant', 94),
('melt', 94),
('couldnt', 93),
('dinner', 93),
('gourmet', 93),
('bear', 93),
('insid', 92),
('uniqu', 92),
('deal', 92),
('night', 92),
('wrap', 92),
('refresh', 92),
('appl', 92),
('onlin', 92),
('book', 91),
('along', 91),
('daughter', 91),
('provid', 91),
('sold', 91),
('mind', 91),
('similar', 91),
('result', 91),
('christma', 90),
('veget', 90),
('site', 90),
('fish', 90),
('feed', 90),
('delight', 89),
('wasnt', 89),
('wine', 89),
('later', 88),
('lover', 88),
('took', 88),
('premium', 88),
('popular', 88),
```

```
('content', 88),
('granola', 88),
('son', 87),
('extrem', 87),
('except', 87),
('youv', 87),
('impress', 87),
('coat', 86),
('conveni', 86),
('wouldnt', 85),
('consum', 85),
('anni', 85),
('set', 84),
('level', 84),
('paper', 84),
('line', 84),
('area', 84),
('loos', 84),
('mint', 84),
('havent', 83),
('american', 83),
('slice', 83),
('classic', 82),
('today', 82),
('label', 82),
('cannot', 82),
('beverag', 82),
('beat', 82),
('hint', 82),
('daili', 82),
('pour', 82),
('gram', 82),
('fair', 82),
('formula', 82),
('word', 81),
('show', 81),
('discov', 81),
('ounc', 81),
('delic', 81),
('sour', 81),
('websit', 81),
('children', 80),
('date', 80),
('search', 80),
('finish', 80),
('kona', 80),
('strawberri', 80),
('salti', 80),
('extract', 80),
('oatmeal', 80),
('heart', 80),
('idea', 79),
('manufactur', 79),
('teeth', 79),
('boil', 79),
('almond', 79),
('crunch', 79),
('grain', 79),
('easier', 78),
('unlik', 78),
('thick', 78),
('sourc', 78),
('lunch', 78),
('cheddar', 78),
('describ', 78),
('preserv', 78),
('possibl', 77),
('creat', 77),
('miss', 77),
('inform', 77),
('pepsi', 77),
('italian', 76),
('thin', 76),
('berri', 76),
('chewi', 76),
('dip', 76),
('warm', 76),
```

```
('given', 75),
('alreadi', 75),
('toast', 75),
('web', 75),
('grow', 75),
('replac', 75),
('potato', 75),
('oat', 75),
('coconut', 75),
('soon', 74),
('play', 74),
('seen', 74),
('caus', 74),
('togeth', 74),
('happen', 74),
('throw', 74),
('huge', 74),
('follow', 73),
('benefit', 73),
('improv', 73),
('system', 73),
('banana', 73),
('child', 72),
('guess', 72),
('beer', 72),
('supermarket', 72),
('smaller', 72),
('blue', 72),
('ate', 72),
('check', 72),
('stevia', 72),
('plan', 71),
('style', 71),
('cover', 71),
('glass', 71),
('arent', 71),
('babi', 71),
('mac', 71),
('someon', 71),
('french', 70),
('difficult', 70),
('wild', 70),
('subtl', 70),
('leaf', 70),
('overal', 70),
('incred', 69),
('hit', 69),
('mom', 69),
('raw', 69),
('clear', 69),
('south', 69),
('macaroni', 69),
('fri', 68),
('digest', 68),
('burton', 67),
('move', 67),
('increas', 67),
('there', 67),
('saw', 67),
('larger', 67),
('eaten', 67),
('stomach', 67),
('unless', 67),
('dessert', 66),
('parti', 66),
('mention', 66),
('wast', 66),
('requir', 66),
('onion', 66),
('rebecca', 66),
('tini', 66),
('stir', 66),
('blood', 66),
('done', 65),
('wow', 65),
('medium', 65),
('deliv', 65)
```

```
('deliv', 65),
('due', 65),
('certifi', 65),
('sandwich', 65),
('bubbl', 65),
('relax', 65),
('scent', 65),
('beetlejuic', 64),
('touch', 64),
('remov', 64),
('imagin', 64),
('popcorn', 64),
('head', 64),
('earli', 64),
('basic', 64),
('standard', 64),
('authent', 64),
('share', 64),
('condit', 63),
('decaf', 63),
('steep', 63),
('stronger', 63),
('gold', 63),
('lime', 63),
('assort', 63),
('chines', 63),
('busi', 63),
('hold', 63),
('avoid', 62),
('within', 62),
('ball', 62),
('doubl', 62),
('cheaper', 62),
('lose', 62),
('garlic', 62),
('homemad', 62),
('instruct', 62),
('yogurt', 62),
('known', 61),
('outstand', 61),
('write', 61),
('sound', 61),
('immedi', 61),
('jelli', 61),
('number', 61),
('visit', 61),
('power', 61),
('medicin', 61),
('juici', 61),
('raspberri', 61),
('twine', 61),
('kraft', 61),
('warn', 60),
('opinion', 60),
('return', 60),
('afternoon', 60),
('sent', 60),
('worri', 60),
('wife', 60),
('rest', 60),
('cheap', 60),
('switch', 60),
('sort', 60),
('perhap', 60),
('option', 60),
('tofu', 60),
('rose', 60),
('tablespoon', 60),
('vet', 60),
('lower', 60),
('crave', 60),
('send', 59),
('bottom', 59),
('heaven', 59),
('told', 59),
('main', 59),
('tend', 59),
('shell', 59),
```

```
('shell', 59),
('hook', 59),
('whatev', 58),
('gone', 58),
('guy', 58),
('figur', 58),
('lost', 58),
('drop', 58),
('thai', 58),
('browni', 58),
('crisp', 58),
('sooth', 58),
('litter', 58),
('pet', 58),
('matter', 57),
('unfortun', 57),
('poor', 57),
('healthier', 57),
('mill', 57),
('age', 56),
('refriger', 56),
('yum', 56),
('san', 56),
('grill', 56),
('over', 56),
('superior', 56),
('caramel', 56),
('veggi', 56),
('reduc', 56),
('deep', 55),
('burn', 55),
('plastic', 55),
('form', 55),
('trip', 55),
('sorri', 55),
('celesti', 55),
('summer', 54),
('break', 54),
('mother', 54),
('restaur', 54),
('kitchen', 54),
('peppermint', 54),
('yes', 54),
('alon', 54),
('chemic', 54),
('allow', 54),
('aftertast', 54),
('broth', 54),
('tin', 54),
('anymor', 54),
('picki', 54),
('anywher', 54),
('continu', 54),
('school', 53),
('agre', 53),
('offic', 53),
('anyway', 53),
('everyday', 53),
('tuna', 53),
('learn', 52),
('introduc', 52),
('frost', 52),
('upon', 52),
('remind', 52),
('bone', 52),
('realiz', 52),
('sprinkl', 52),
('tart', 52),
('pancak', 52),
('left', 52),
('fit', 52),
('util', 52),
('previous', 51),
('round', 51),
('hate', 51),
('sampl', 51),
('carbon', 51),
('drinker', 51)
```

```
('drinker', 51),
('outsid', 51),
('hershey', 51),
('spoon', 51),
('sip', 51),
('shake', 51),
('muffin', 51),
('kitti', 51),
('douw', 51),
('lipton', 51),
('pan', 50),
('remain', 50),
('control', 50),
('job', 50),
('countri', 50),
('spread', 50),
('otherwis', 50),
('count', 50),
('readi', 50),
('none', 50),
('egbert', 50),
('vegan', 50),
('typic', 50),
('bland', 50),
('moist', 50),
('hungri', 50),
('allergi', 50),
('crazi', 49),
('appear', 49),
('present', 49),
('sale', 49),
('smoke', 49),
('decent', 49),
('eye', 49),
('chunk', 49),
('nutti', 49),
('flake', 49),
('success', 48),
('strang', 48),
('liquid', 48),
('intens', 48),
('pictur', 48),
('room', 48),
('distinct', 48),
('what', 48),
('yellow', 48),
('salmon', 48),
('shelf', 48),
('heavi', 48),
('chamomil', 48),
('brought', 48),
('biscuit', 48),
('citrus', 48),
('xylitol', 48),
('tim', 47),
('stori', 47),
('steak', 47),
('claim', 47),
('depend', 47),
('overpow', 47),
('straight', 47),
('deliveri', 47),
('bonsai', 47),
('cane', 47),
('safe', 47),
('raisin', 47),
('toy', 47),
('yeast', 47),
('quantiti', 47),
('somewhat', 47),
('tip', 47),
('funni', 46),
('mess', 46),
('tabl', 46),
('china', 46),
('serious', 46),
('suppos', 46),
('island', 46),
```

```
('island', 46),
('grown', 46),
('guest', 46),
('mountain', 46),
('infus', 46),
('chili', 46),
('weve', 46),
('golden', 46),
('mug', 46),
('oreo', 46),
('fabul', 45),
('terrif', 45),
('short', 45),
('relat', 45),
('appreci', 45),
('whether', 45),
('salsa', 45),
('equal', 45),
('crumbl', 45),
('began', 45),
('hazelnut', 45),
('darjeel', 45),
('stale', 45),
('stress', 45),
('tooth', 45),
('boy', 45),
('heard', 45),
('kosher', 45),
('kashi', 45),
('spend', 44),
('suppli', 44),
('magic', 44),
('worst', 44),
('knew', 44),
('twice', 44),
('six', 44),
('jasmin', 44),
('paid', 44),
('pie', 44),
('hibiscus', 44),
('bulk', 44),
('basket', 44),
('respons', 44),
('fall', 44),
('forward', 44),
('supplement', 44),
('cent', 44),
('dollar', 44),
('mate', 44),
('kick', 44),
('kept', 44),
('current', 44),
('quot', 44),
('hes', 43),
('air', 43),
('design', 43),
('terribl', 43),
('stuck', 43),
('hear', 43),
('sick', 43),
('pleasur', 43),
('sens', 43),
('stand', 43),
('talk', 43),
('horribl', 43),
('complex', 43),
('curri', 43),
('charg', 43),
('acquir', 43),
...]
```

In [79]:

```
type(word_dist)
```

Out[79]:

list

In [80]:

```
word_dist[:15]
```

Out[80]:

```
[('tea', 2662),
 ('tast', 2442),
 ('like', 2392),
 ('flavor', 1996),
 ('good', 1871),
 ('one', 1738),
 ('great', 1718),
 ('use', 1634),
 ('tri', 1511),
 ('make', 1440),
 ('product', 1395),
 ('love', 1291),
 ('coffe', 1247),
 ('get', 1243),
 ('best', 1192)]
```

In [0]:

```
#https://stackoverflow.com/questions/3071415/efficient-method-to-calculate-the-rank-vector-of-a-li
st-in-python


'''a={}
  rank=1
  for num in sorted(vector):
    if num not in a:
      a[num]=rank
      rank=rank+1 '''

a = {}
rank = 1
for num in range(len(word_dist)):
    i = word_dist[num][0]
    a[i] = rank
    rank+=1
```

In [0]:

```
X = []
for sent in z:
    rows = []
    for word in sent.split():
        rows.append(a[word])
    X.append(rows)
```

In [0]:

```
#X[:20]
```

In [97]:

```
X = np.array(X)
X
```

Out[97]:

```
array([list([110, 527, 7190, 26, 527, 10, 550, 1438, 3537, 5395, 1229, 1230, 528, 113, 1943, 5396,
985, 837, 5397, 1412, 4472, 12, 130, 582, 527, 838, 2450, 570, 527, 1520, 2009, 550, 85, 335,
5395, 1016, 1376]),
       list([119, 2983, 7, 63, 961, 18, 7191, 133, 583, 986, 2984, 39, 280, 647, 1521, 543, 155, 34
, 3240, 129, 8, 2983, 527, 2010, 1413, 2192, 5398, 297, 119, 831, 159, 528, 527, 589, 346, 219]),
       list([7192, 7193, 28, 1861, 288, 376, 88, 1350, 308, 433, 540, 3241, 2011, 288]),
       ...,
```

```
       list([252, 265, 16038, 9, 3909, 990, 746, 2561, 1214, 1210, 64, 385, 818, 1167, 8, 3909, 10,
283, 2079, 156, 5027, 16039, 16040, 3909, 8, 265, 178, 223, 64, 51, 536, 8, 3861, 4, 1129, 4201, 1
560, 3909, 17, 8, 1008, 44, 1423, 567, 9, 3909, 3286, 363, 1243, 8, 1021, 1214]),
       list([39, 188, 16, 16041, 188, 852, 1815, 159, 2304, 468, 174, 629, 113, 408, 1815, 396,
216, 14, 16, 312, 12, 87, 174, 856, 174, 204, 1815, 414, 4396, 249, 85, 7, 52, 479, 480, 17, 7, 15
7, 757, 630, 484, 1516, 254, 2304, 260, 77, 204, 2272, 1815, 174, 232, 1596, 743, 757, 630, 484, 1
079, 130, 484, 1079, 933, 249, 4396, 29, 31, 50, 1079, 775, 254, 112, 216, 479, 480, 21, 54, 446,
256, 554, 10, 195]),
       list([474, 16042, 371, 252, 20, 544, 637, 2704, 608, 474, 210, 29, 648, 91, 81, 71, 268, 28
, 25, 358])],
      dtype=object)
```

In [98]:

```
Y = final['Score']
Y = np.array(Y)
Y
```

Out[98]:

```
array([1, 1, 1, ..., 1, 1, 1])
```

In [0]:

```
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 = 0,shuffle
= False)
```

In [0]:

```
%matplotlib notebook
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import time
# https://gist.github.com/greydanus/f6eee59eaf1d90fcb3b534a25362cea4
# https://stackoverflow.com/a/14434334
# this function is used to update the plots for each epoch and error
def plt_dynamic(x, vy, ty, ax, colors=['b']):
    ax.plot(x, vy, 'b', label="Validation Loss")
    ax.plot(x, ty, 'r', label="Train Loss")
    plt.legend()
    plt.grid()
    fig.canvas.draw()
```

In [0]:

```
import warnings
plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = [10, 5]
warnings.filterwarnings("ignore", category=FutureWarning)
%config InlineBackend.figure_format = 'retina'
```

In [104]:

```
# truncate and/or pad input sequences
max_review_length = 600
X_train = sequence.pad_sequences(X_train, maxlen=max_review_length)
X_test = sequence.pad_sequences(X_test, maxlen=max_review_length)

print(X_train.shape)
print(X_train[1])
```

```
(4000, 600)
[    0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0  119 2983    7   63  961   18 7191  133  583  986
 2984   39  280  647 1521  543  155   34 3240  129    8 2983  527 2010
 1413 2192 5398  297  119  831  159  528  527  589  346  219]
```

In [0]:

```
top_words = 6000
epochs = 20
batch_size = 64
embedding_vecor_length = 32
```

In [0]:

```
from keras.layers.normalization import BatchNormalization
```

In [0]:

```
from datetime import datetime
```

Architecture-1

In [122]:

```
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
model.add(LSTM(2, return_sequences=True))
model.add(LSTM(2))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_26 (Embedding)     (None, 600, 32)           192000    
_________________________________________________________________
lstm_51 (LSTM)               (None, 600, 2)            280       
```

```
_________________________________________________________________
lstm_52 (LSTM)               (None, 2)                 40
_________________________________________________________________
dense_21 (Dense)             (None, 1)                 3
=================================================================
Total params: 192,323
Trainable params: 192,323
Non-trainable params: 0
_________________________________________________________________
None
```

In [123]:

```
start = datetime.now()


history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))
```

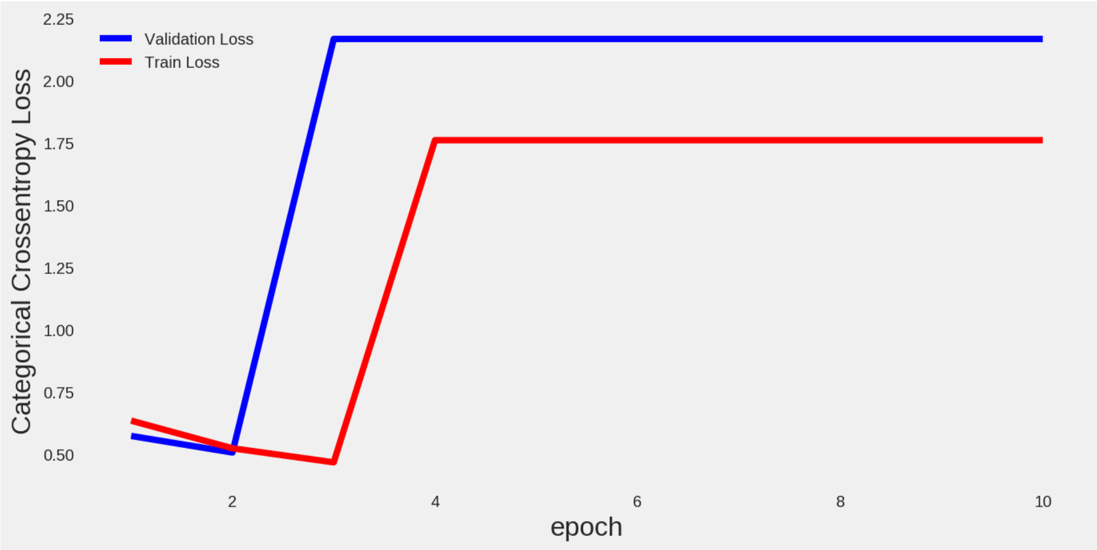
```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 178s 45ms/step - loss: 0.6351 - acc: 0.8855 - val_los
s: 0.5737 - val_acc: 0.8640
Epoch 2/10
4000/4000 [==============================] - 172s 43ms/step - loss: 0.5241 - acc: 0.8895 - val_los
s: 0.5071 - val_acc: 0.8640
Epoch 3/10
4000/4000 [==============================] - 175s 44ms/step - loss: 0.4677 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 4/10
4000/4000 [==============================] - 178s 45ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 5/10
4000/4000 [==============================] - 186s 46ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 6/10
4000/4000 [==============================] - 174s 44ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 7/10
4000/4000 [==============================] - 176s 44ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 8/10
4000/4000 [==============================] - 178s 45ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 9/10
4000/4000 [==============================] - 175s 44ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Epoch 10/10
4000/4000 [==============================] - 178s 45ms/step - loss: 1.7616 - acc: 0.8895 - val_los
s: 2.1682 - val_acc: 0.8640
Accuracy: 86.40%
```

In [124]:

```
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,11))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs
```

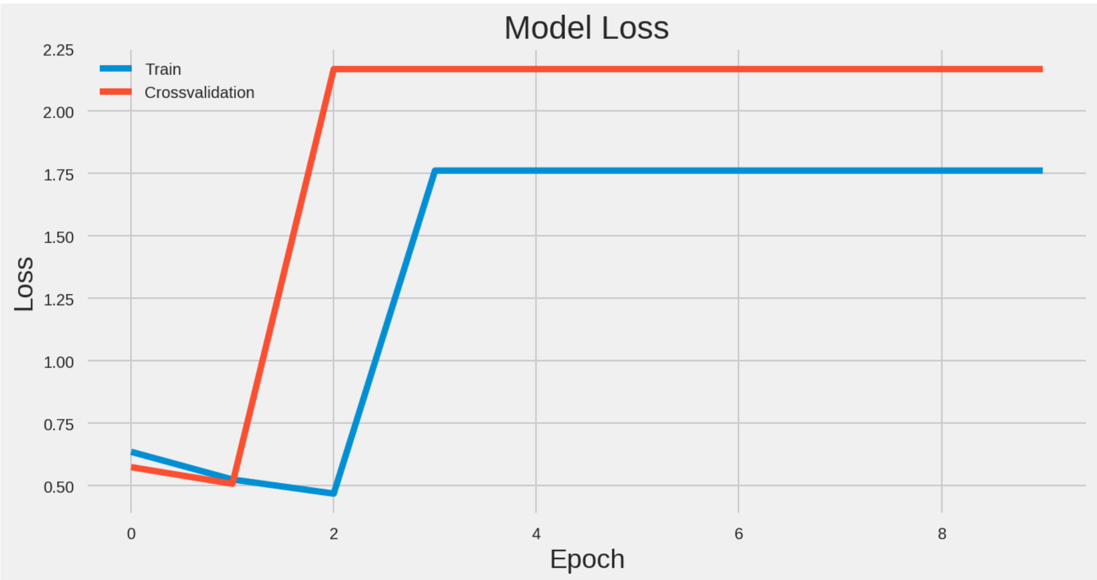
```
vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)


print('Time taken to run this cell :', datetime.now() - start)
```

Time taken to run this cell : 0:30:12.316704



In [125]:

```
plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'], loc='upper left')
plt.show()
```



Architecture-2

In [126]:

```
from keras.layers.normalization import BatchNormalization


# create the model
embedding_vecor_length = 64
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
```

```
model.add(BatchNormalization())
model.add(LSTM(100, return_sequences=True))
model.add(Dropout(0.25))
model.add(BatchNormalization())
model.add(LSTM(100))
model.add(Dropout(0.25))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
#Refer: https://datascience.stackexchange.com/questions/10615/number-of-parameters-in-an-lstm-mode
l
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_27 (Embedding)     (None, 600, 64)           384000    
_________________________________________________________________
batch_normalization_37 (Batc (None, 600, 64)           256       
_________________________________________________________________
lstm_53 (LSTM)               (None, 600, 100)          66000     
_________________________________________________________________
dropout_40 (Dropout)         (None, 600, 100)          0         
_________________________________________________________________
batch_normalization_38 (Batc (None, 600, 100)          400       
_________________________________________________________________
lstm_54 (LSTM)               (None, 100)               80400     
_________________________________________________________________
dropout_41 (Dropout)         (None, 100)               0         
_________________________________________________________________
dense_22 (Dense)             (None, 1)                 101       
=================================================================
Total params: 531,157
Trainable params: 530,829
Non-trainable params: 328
_________________________________________________________________
None
```

In [129]:

```
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'], loc='upper right')
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 171s 43ms/step - loss: 0.1379 - acc: 0.9490 - val_los
s: 0.3757 - val_acc: 0.8620
Epoch 2/10
4000/4000 [==============================] - 176s 44ms/step - loss: 0.1352 - acc: 0.9510 - val_los
s: 0.4866 - val_acc: 0.8810
Epoch 3/10
4000/4000 [==============================] - 183s 46ms/step - loss: 0.0831 - acc: 0.9698 - val_los
s: 0.4728 - val_acc: 0.8840
Epoch 4/10
4000/4000 [==============================] - 174s 44ms/step - loss: 0.0446 - acc: 0.9862 - val_los
s: 0.5302 - val_acc: 0.8510
Epoch 5/10
4000/4000 [==============================] - 176s 44ms/step - loss: 0.0365 - acc: 0.9875 - val_los
s: 0.6105 - val_acc: 0.8760
Epoch 6/10
4000/4000 [==============================] - 176s 44ms/step - loss: 0.0551 - acc: 0.9800 - val_los
```

```
s: 0.4866 - val_acc: 0.8710
Epoch 7/10
4000/4000 [==============================] - 182s 46ms/step - loss: 0.0496 - acc: 0.9840 - val_los
s: 0.5046 - val_acc: 0.8690
Epoch 8/10
4000/4000 [==============================] - 180s 45ms/step - loss: 0.0466 - acc: 0.9845 - val_los
s: 0.5959 - val_acc: 0.8810
Epoch 9/10
4000/4000 [==============================] - 181s 45ms/step - loss: 0.0294 - acc: 0.9915 - val_los
s: 0.6133 - val_acc: 0.8580
Epoch 10/10
4000/4000 [==============================] - 178s 45ms/step - loss: 0.0272 - acc: 0.9908 - val_los
s: 0.6650 - val_acc: 0.8840
Accuracy: 88.40%
```


```
Time taken to run this cell : 0:30:12.283487
```

Architecture-3

In [130]:

```
# create the model
embedding_vecor_length = 64
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
model.add(BatchNormalization())
model.add(LSTM(2, return_sequences=True))
model.add(BatchNormalization())
model.add(LSTM(2))
model.add(Dropout(0.25))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
#Refer: https://datascience.stackexchange.com/questions/10615/number-of-parameters-in-an-lstm-mode
l
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_28 (Embedding)     (None, 600, 64)           384000
_________________________________________________________________
batch_normalization_39 (Batc (None, 600, 64)           256
_________________________________________________________________
lstm_55 (LSTM)               (None, 600, 2)            536
_________________________________________________________________
batch_normalization_40 (Batc (None, 600, 2)            8
_________________________________________________________________
lstm_56 (LSTM)               (None, 2)                 40
_________________________________________________________________
dropout_42 (Dropout)         (None, 2)                 0
_________________________________________________________________
```

```
dense_23 (Dense)             (None, 1)                 3
=================================================================
Total params: 384,843
Trainable params: 384,711
Non-trainable params: 132
_________________________________________________________________
None
```

In [131]:

```
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'])
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 181s 45ms/step - loss: 0.5411 - acc: 0.7970 - val_los
s: 0.4908 - val_acc: 0.8640
Epoch 2/10
4000/4000 [==============================] - 169s 42ms/step - loss: 0.4744 - acc: 0.8895 - val_los
s: 0.4633 - val_acc: 0.8640
Epoch 3/10
4000/4000 [==============================] - 168s 42ms/step - loss: 0.4495 - acc: 0.8895 - val_los
s: 0.4483 - val_acc: 0.8640
Epoch 4/10
4000/4000 [==============================] - 168s 42ms/step - loss: 0.4335 - acc: 0.8895 - val_los
s: 0.4300 - val_acc: 0.8640
Epoch 5/10
4000/4000 [==============================] - 169s 42ms/step - loss: 0.4036 - acc: 0.8895 - val_los
s: 0.4124 - val_acc: 0.8640
Epoch 6/10
4000/4000 [==============================] - 170s 42ms/step - loss: 0.3771 - acc: 0.8895 - val_los
s: 0.3957 - val_acc: 0.8640
Epoch 7/10
4000/4000 [==============================] - 169s 42ms/step - loss: 0.3482 - acc: 0.8900 - val_los
s: 0.3837 - val_acc: 0.8640
Epoch 8/10
4000/4000 [==============================] - 169s 42ms/step - loss: 0.3278 - acc: 0.9000 - val_los
s: 0.3856 - val_acc: 0.8620
Epoch 9/10
4000/4000 [==============================] - 170s 42ms/step - loss: 0.3018 - acc: 0.9073 - val_los
s: 0.3786 - val_acc: 0.8540
Epoch 10/10
4000/4000 [==============================] - 170s 42ms/step - loss: 0.2922 - acc: 0.9113 - val_los
s: 0.3676 - val_acc: 0.8600
Accuracy: 86.00%
```


```
Time taken to run this cell : 0:28:58.181136
```

Architecture-4

In [136]:

```
# create the model
embedding_vecor_length = 32
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
model.add(LSTM(100))
model.add(Dropout(0.25))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
#Refer: https://datascience.stackexchange.com/questions/10615/number-of-parameters-in-an-lstm-mode
l
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_31 (Embedding)     (None, 600, 32)           192000    
_________________________________________________________________
lstm_59 (LSTM)               (None, 100)               53200     
_________________________________________________________________
dropout_45 (Dropout)         (None, 100)               0         
_________________________________________________________________
dense_26 (Dense)             (None, 1)                 101       
=================================================================
Total params: 245,301
Trainable params: 245,301
Non-trainable params: 0
_________________________________________________________________
None
```

In [137]:

```
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'])
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 100s 25ms/step - loss: 0.3988 - acc: 0.8812 - val_los
s: 0.3843 - val_acc: 0.8640
Epoch 2/10
4000/4000 [==============================] - 88s 22ms/step - loss: 1.7287 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 3/10
4000/4000 [==============================] - 87s 22ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
```

```
: 2.1682 - val_acc: 0.8640
Epoch 4/10
4000/4000 [==============================] - 88s 22ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 5/10
4000/4000 [==============================] - 90s 23ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 6/10
4000/4000 [==============================] - 91s 23ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 7/10
4000/4000 [==============================] - 90s 23ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 8/10
4000/4000 [==============================] - 93s 23ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 9/10
4000/4000 [==============================] - 94s 24ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Epoch 10/10
4000/4000 [==============================] - 91s 23ms/step - loss: 1.7616 - acc: 0.8895 - val_loss
: 2.1682 - val_acc: 0.8640
Accuracy: 86.40%
```


```
Time taken to run this cell : 0:15:33.636212
```

Architecture-5

In [149]:

```
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
model.add(LSTM(10,return_sequences=True,dropout=0.5,recurrent_dropout=0.5))
model.add(LSTM(10,dropout=0.5,recurrent_dropout=0.5))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
print(model.summary())
```

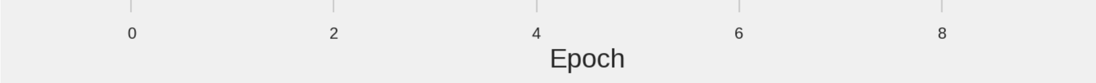
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_37 (Embedding)     (None, 600, 32)           192000
_________________________________________________________________
lstm_71 (LSTM)               (None, 600, 10)           1720
_________________________________________________________________
lstm_72 (LSTM)               (None, 10)                840
_________________________________________________________________
dense_31 (Dense)             (None, 1)                 11
=================================================================
Total params: 194,571
Trainable params: 194,571
```

```
Non-trainable params: 0
_________________________________________________________________
None
```

In [150]:

```
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'])
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 216s 54ms/step - loss: 0.5445 - acc: 0.8738 - val_los
s: 0.3943 - val_acc: 0.8640
Epoch 2/10
4000/4000 [==============================] - 199s 50ms/step - loss: 0.3490 - acc: 0.8892 - val_los
s: 0.4105 - val_acc: 0.8640
Epoch 3/10
4000/4000 [==============================] - 196s 49ms/step - loss: 0.3330 - acc: 0.8895 - val_los
s: 0.3932 - val_acc: 0.8640
Epoch 4/10
4000/4000 [==============================] - 193s 48ms/step - loss: 0.3144 - acc: 0.8900 - val_los
s: 0.3665 - val_acc: 0.8690
Epoch 5/10
4000/4000 [==============================] - 192s 48ms/step - loss: 0.2662 - acc: 0.9008 - val_los
s: 0.3734 - val_acc: 0.8680
Epoch 6/10
4000/4000 [==============================] - 194s 48ms/step - loss: 0.2406 - acc: 0.9113 - val_los
s: 0.3701 - val_acc: 0.8690
Epoch 7/10
4000/4000 [==============================] - 191s 48ms/step - loss: 0.2164 - acc: 0.9248 - val_los
s: 0.3909 - val_acc: 0.8670
Epoch 8/10
4000/4000 [==============================] - 192s 48ms/step - loss: 0.2051 - acc: 0.9223 - val_los
s: 0.4047 - val_acc: 0.8650
Epoch 9/10
4000/4000 [==============================] - 193s 48ms/step - loss: 0.1817 - acc: 0.9315 - val_los
s: 0.4272 - val_acc: 0.8670
Epoch 10/10
4000/4000 [==============================] - 190s 47ms/step - loss: 0.1777 - acc: 0.9345 - val_los
s: 0.4367 - val_acc: 0.8670
Accuracy: 86.70%
```




```
Time taken to run this cell : 0:33:14.242946
```

Architecture-6

In [159]:

```python
model = Sequential()
model.add(Embedding(top_words, embedding_vecor_length, input_length=max_review_length))
model.add(LSTM(2,return_sequences=True,dropout=0.5,recurrent_dropout=0.5))
model.add(LSTM(2,dropout=0.5,recurrent_dropout=0.5))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
print(model.summary())
```

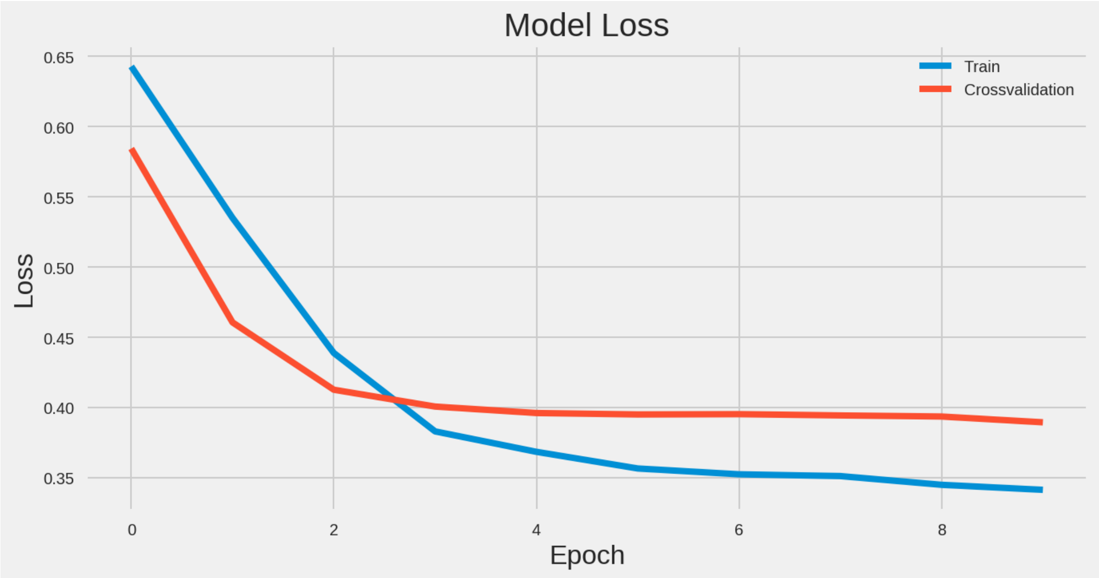
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_41 (Embedding)     (None, 600, 64)           384000    
_________________________________________________________________
lstm_79 (LSTM)               (None, 600, 2)            536       
_________________________________________________________________
lstm_80 (LSTM)               (None, 2)                 40        
_________________________________________________________________
dense_35 (Dense)             (None, 1)                 3         
=================================================================
Total params: 384,579
Trainable params: 384,579
Non-trainable params: 0
_________________________________________________________________
None
```

In [152]:

```python
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 10, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'])
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/10
4000/4000 [==============================] - 211s 53ms/step - loss: 0.6429 - acc: 0.8550 - val_los
s: 0.5844 - val_acc: 0.8640
Epoch 2/10
4000/4000 [==============================] - 192s 48ms/step - loss: 0.5350 - acc: 0.8870 - val_los
s: 0.4606 - val_acc: 0.8640
Epoch 3/10
4000/4000 [==============================] - 194s 48ms/step - loss: 0.4388 - acc: 0.8892 - val_los
s: 0.4126 - val_acc: 0.8640
Epoch 4/10
4000/4000 [==============================] - 194s 48ms/step - loss: 0.3829 - acc: 0.8895 - val_los
s: 0.4006 - val_acc: 0.8640
Epoch 5/10
4000/4000 [==============================] - 193s 48ms/step - loss: 0.3684 - acc: 0.8898 - val_los
s: 0.3960 - val_acc: 0.8640
Epoch 6/10
4000/4000 [==============================] - 196s 49ms/step - loss: 0.3565 - acc: 0.8895 - val_los
```

```
4000/4000 [==============================] - 196s 49ms/step - loss: 0.3565 - acc: 0.8895 - val_los
s: 0.3950 - val_acc: 0.8640
Epoch 7/10
4000/4000 [==============================] - 200s 50ms/step - loss: 0.3524 - acc: 0.8892 - val_los
s: 0.3952 - val_acc: 0.8640
Epoch 8/10
4000/4000 [==============================] - 202s 50ms/step - loss: 0.3511 - acc: 0.8892 - val_los
s: 0.3943 - val_acc: 0.8640
Epoch 9/10
4000/4000 [==============================] - 202s 51ms/step - loss: 0.3449 - acc: 0.8892 - val_los
s: 0.3935 - val_acc: 0.8640
Epoch 10/10
4000/4000 [==============================] - 201s 50ms/step - loss: 0.3413 - acc: 0.8898 - val_los
s: 0.3894 - val_acc: 0.8640
Accuracy: 86.40%
```




```
Time taken to run this cell : 0:33:45.448443
```

Architecture-7

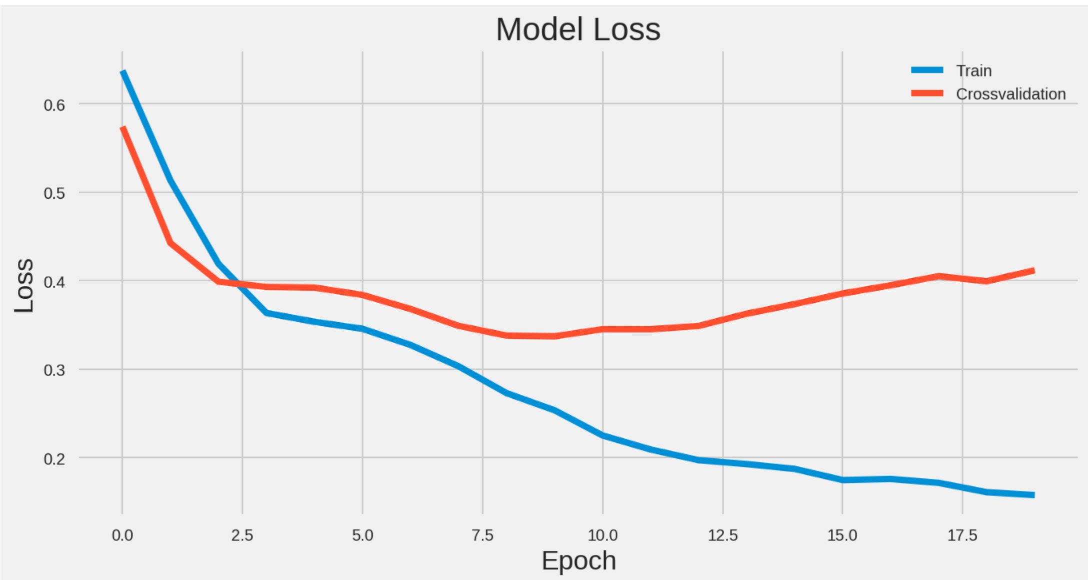
(Architecture-6 with 20 epochs.)

In [161]:

```
start = datetime.now()

history = model.fit(X_train, y_train, epochs = 20, batch_size = batch_size,
verbose=1,validation_data=(X_test, y_test))
scores = model.evaluate(X_test, y_test, verbose = 0)
print("Accuracy: %.2f%%" % (scores[1]*100))



plt.plot(model.history.history['loss'])
plt.plot(model.history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Crossvalidation'])
plt.show()
print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 4000 samples, validate on 1000 samples
Epoch 1/20
4000/4000 [==============================] - 215s 54ms/step - loss: 0.6373 - acc: 0.8725 - val_los
s: 0.5740 - val_acc: 0.8640
Epoch 2/20
4000/4000 [==============================] - 199s 50ms/step - loss: 0.5132 - acc: 0.8895 - val_los
s: 0.4425 - val_acc: 0.8640
Epoch 3/20
4000/4000 [==============================] - 200s 50ms/step - loss: 0.4190 - acc: 0.8895 - val_los
s: 0.3988 - val_acc: 0.8640
```

```
Epoch 4/20
4000/4000 [==============================] - 201s 50ms/step - loss: 0.3635 - acc: 0.8895 - val_los
s: 0.3929 - val_acc: 0.8640
Epoch 5/20
4000/4000 [==============================] - 203s 51ms/step - loss: 0.3535 - acc: 0.8895 - val_los
s: 0.3922 - val_acc: 0.8640
Epoch 6/20
4000/4000 [==============================] - 204s 51ms/step - loss: 0.3456 - acc: 0.8895 - val_los
s: 0.3839 - val_acc: 0.8640
Epoch 7/20
4000/4000 [==============================] - 207s 52ms/step - loss: 0.3274 - acc: 0.8898 - val_los
s: 0.3680 - val_acc: 0.8640
Epoch 8/20
4000/4000 [==============================] - 201s 50ms/step - loss: 0.3033 - acc: 0.8907 - val_los
s: 0.3490 - val_acc: 0.8640
Epoch 9/20
4000/4000 [==============================] - 195s 49ms/step - loss: 0.2730 - acc: 0.9008 - val_los
s: 0.3380 - val_acc: 0.8640
Epoch 10/20
4000/4000 [==============================] - 197s 49ms/step - loss: 0.2536 - acc: 0.9040 - val_los
s: 0.3371 - val_acc: 0.8660
Epoch 11/20
4000/4000 [==============================] - 199s 50ms/step - loss: 0.2252 - acc: 0.9183 - val_los
s: 0.3452 - val_acc: 0.8710
Epoch 12/20
4000/4000 [==============================] - 200s 50ms/step - loss: 0.2093 - acc: 0.9183 - val_los
s: 0.3451 - val_acc: 0.8710
Epoch 13/20
4000/4000 [==============================] - 198s 49ms/step - loss: 0.1972 - acc: 0.9275 - val_los
s: 0.3488 - val_acc: 0.8700
Epoch 14/20
4000/4000 [==============================] - 196s 49ms/step - loss: 0.1928 - acc: 0.9278 - val_los
s: 0.3627 - val_acc: 0.8710
Epoch 15/20
4000/4000 [==============================] - 194s 49ms/step - loss: 0.1874 - acc: 0.9313 - val_los
s: 0.3735 - val_acc: 0.8730
Epoch 16/20
4000/4000 [==============================] - 195s 49ms/step - loss: 0.1748 - acc: 0.9365 - val_los
s: 0.3855 - val_acc: 0.8710
Epoch 17/20
4000/4000 [==============================] - 196s 49ms/step - loss: 0.1760 - acc: 0.9365 - val_los
s: 0.3948 - val_acc: 0.8680
Epoch 18/20
4000/4000 [==============================] - 197s 49ms/step - loss: 0.1716 - acc: 0.9387 - val_los
s: 0.4051 - val_acc: 0.8670
Epoch 19/20
4000/4000 [==============================] - 196s 49ms/step - loss: 0.1611 - acc: 0.9415 - val_los
s: 0.3993 - val_acc: 0.8640
Epoch 20/20
4000/4000 [==============================] - 196s 49ms/step - loss: 0.1578 - acc: 0.9400 - val_los
s: 0.4117 - val_acc: 0.8660
Accuracy: 86.60%
```




```
Time taken to run this cell : 1:07:04.966635
```

Summary

As my previous models were overfitting highly, i decided to try different architechures. But due to time limitations, I had to do it with 5000 points only(Sorry!!!!!!...). Here is a ocmplete summary of the 7 architures that i used.

In [11]:

```
from prettytable import PrettyTable


x = PrettyTable()

x.field_names = ["S.R",'Architectures',"Training Loss","Test loss","Accuracy","Remarks"]

x.add_row([(1),"Architecture-1","1.716", 2.162, '86.40%', "Overfit"])
x.add_row([(2),"Architecture-2","0.0272", 0.9908, '88.40%', "Overfit"])
x.add_row([(3),'Architecture-3',"0.2922", 0.3676, '86.00%', "Good"])
x.add_row([(4),"Architecture-4","1.7616", 2.1682, '86.40%', "Overfit"])
x.add_row([(5),"Architecture-5","0.1777" , 0.4367, '86.70%',"Good"])
x.add_row([(6),'Architecture-6',"0.3413", 0.3894, '86.40%', "Very Good"])
x.add_row([(7),'Architecture-7(Architecture-6 with 20 epochs.)',"0.1578", 0.4117, '86.60%', "Good"]
)
print(x.get_string(title = "-----SUMMARY-----"))
```

```
+-----+-------------------------------------------------+---------------+-----------+----------+----
----+
| S.R |                  Architectures                  | Training Loss | Test loss | Accuracy |  Re
arks  |
+-----+-------------------------------------------------+---------------+-----------+----------+----
----+
|  1  |                  Architecture-1                 |     1.716     |   2.162   |  86.40%  |  Ov
fit  |
|  2  |                  Architecture-2                 |     0.0272    |   0.9908  |  88.40%  |
Overfit   |
|  3  |                  Architecture-3                 |     0.2922    |   0.3676  |  86.00%  |
Good    |
|  4  |                  Architecture-4                 |     1.7616    |   2.1682  |  86.40%  |
Overfit   |
|  5  |                  Architecture-5                 |     0.1777    |   0.4367  |  86.70%  |
Good    |
|  6  |                  Architecture-6                 |     0.3413    |   0.3894  |  86.40%  | Ver
Good |
|  7  | Architecture-7(Architecture-6 with 20 epochs.) |     0.1578    |   0.4117  |  86.60%  |
Good    |
+-----+-------------------------------------------------+---------------+-----------+----------+----
----+
```