In [1]:
```
import os
import pandas as pd
import re
import string

from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
import gensim
```

In [2]:
```
def preprocessing_corpus(corpus):

    '''
    Takes string from news corpus, normalizes and tokenizes into sentenc
es
    '''


    # Removal of white space
    step_1 = re.sub('\s+', ' ', corpus)
    # RFrom/To and subject line of email
    step_2 = re.sub(r'\bFrom: .*? writes: ', '', step_1)
    # Removal of digits
    step_3 = re.sub(r'\d+', '', step_2)
    # Remove other signs
    step_4 = re.sub(r'["_“”\'\`\-\*\(\)]', '', step_3)
    # Tokenize to sentences while punctuation is still in place
    tokens_news = sent_tokenize(step_4)
    # Convert each token to lowercase
    lower_token = list(map(lambda token: token.lower(), tokens_news))
    # Remove punctuation from lowercase token
    punct_less_token = list(map(lambda token:
                                token.translate(str.maketrans('', '', st
ring.punctuation)), lower_token))

    return punct_less_token


def build_word2vec(list_of_list, dimension_size, window_size, min_obs, m
odel_type, model_name):

    """
    Creates a model object
    Args:
        list_of_list (list): preprocessed text corpus
        dimension_size (int): size of dimensions in model
        window_size (int): window size used for training
        min_obs (int): minimum observed instances of a word to be consid
ered
        model_type (binary 1 or 0): 1 = skipgram, 0 = CBOW
        model_name: name of object
    Returns:
        A trained word2vec model, that is saved as an object
    """

    new_model = gensim.models.Word2Vec(list_of_list, vector_size=dimensi
on_size, window=window_size,
                                       sg=model_type, min_count=min_obs)
    new_model.save(model_name)

    return new_model


def evaluate_model_nn(model_object, index_model_name, evaluation_word_li
sts, top_n):
    """
    Identifies top n nearest words for words in a lit
```

```
    Args:
        model_object (obj): word2vec model
        index_model_name (str): name of model for row indices naming
        evaluation_word_lists (list): words to be used as reference to i
dentify nearest neighbors
        top_n (int): top n number of nearest neighbors
    Returns:
        A pd dataframe summary
    """

    # Create blank df
    results_df = pd.DataFrame()

    for i, word in enumerate(evaluation_word_lists):
        result = model_object.wv.most_similar(word, topn=top_n)
        result = list(map(lambda x: str(x[0]) + ', ' + str(round(x[1],3
)), result))
        results_df[word] = [result]

    results_df.index = [str(index_model_name)]

    return results_df
```

In [3]:
```
# Reading in pre-processed gendered df
gender_mentionned = pd.read_csv('yelp_gendered.csv')
gender_mentionned_unique = gender_mentionned[(gender_mentionned['male_pr
esent'] +
                                              gender_mentionned['female_
present']) == 1]
print("- Gendered CSV read, entries referencing both genders are remove
d")
```

```
- Gendered CSV read, entries referencing both genders are removed
```

In [4]:
```
# Subset of full dataset
subset = gender_mentionned_unique[0:100000]

# Create a list of all words (for word2vec modeling)
word_tokens_list_of_list = []
for review in subset.text:
    tokenized_review = preprocessing_corpus(review)
    for sentence in tokenized_review:
        word_tokens_list = word_tokenize(sentence)
        word_tokens_list_of_list.append(word_tokens_list)
print('- All words of corpus added to list of list')
print('- READY FOR WORD2VEC MODELING!')
```

```
- All words of corpus added to list of list
- READY FOR WORD2VEC MODELING!
```

In [5]:
```
# Create Skipgram model
model_sg_100_5 = build_word2vec(word_tokens_list_of_list, 100, 5, 5, 1,
"model_sg_100_5")
print('+ Created skipgram models with 100 embeddings and window size 5')
```

```
+ Created skipgram models with 100 embeddings and window size 5
```

In [9]:
```
pd.set_option('max_colwidth', 500)
evaluation_list = ['lady', 'sales', 'owner', 'employee', 'cashier', 'dri
ver', 'saleswoman', 'salesman']
agg_results = evaluate_model_nn(model_sg_100_5, 'model_sg_100_5', evalua
tion_list, 10)
agg_results
```

Out[9]:

	lady	sales	owner	employee	cashier	driver	
model_sg_100_5	[woman, 0.939, girl, 0.903, gal, 0.822, gentleman, 0.788, guy, 0.725, gent, 0.711, fella, 0.691, man, 0.676, cashier, 0.673, person, 0.665]	[finance, 0.803, salesperson, 0.735, salesman, 0.711, patrick, 0.7, rep, 0.667, financing, 0.664, leasing, 0.648, cody, 0.642, vito, 0.639, consultants, 0.636]	[manager, 0.803, gm, 0.786, proprietor, 0.773, managerowner, 0.773, ownermanager, 0.773, owners, 0.742, manger, 0.715, chefowner, 0.711, trevor, 0.71, paige, 0.675]	[staffer, 0.814, associate, 0.801, worker, 0.793, clerk, 0.756, cashier, 0.737, wench, 0.732, pharmacist, 0.724, person, 0.717, attendant, 0.702, saleswoman, 0.699]	[register, 0.867, counter, 0.809, clerk, 0.777, barista, 0.74, employee, 0.737, trainee, 0.732, sneered, 0.728, gent, 0.726, staffer, 0.721, greeter, 0.72]	[dispatcher, 0.767, drivers, 0.7, operator, 0.694, shuttle, 0.687, bellman, 0.677, delivery, 0.674, mover, 0.671, dispatch, 0.67, estimator, 0.666, lyft, 0.665]	as bart 0.7 d r sa

In [7]:

```
# Printing results
for i in range(0,agg_results.shape[1]):
    print('NEAREST NEIGHBORS FOR: ' + agg_results.columns[i])
    result = agg_results.iloc[0,i]
    for j in result:
        print(j)
    print('---   ---   ---   ---   ---   ---   ---   ---')
```

NEAREST NEIGHBORS FOR: lady
woman, 0.939
girl, 0.903
gal, 0.822
gentleman, 0.788
guy, 0.725
gent, 0.711
fella, 0.691
man, 0.676
cashier, 0.673
person, 0.665
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: sales
finance, 0.803
salesperson, 0.735
salesman, 0.711
patrick, 0.7
rep, 0.667
financing, 0.664
leasing, 0.648
cody, 0.642
vito, 0.639
consultants, 0.636
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: owner
manager, 0.803
gm, 0.786
proprietor, 0.773
managerowner, 0.773
ownermanager, 0.773
owners, 0.742
manger, 0.715
chefowner, 0.711
trevor, 0.71
paige, 0.675
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: employee
staffer, 0.814
associate, 0.801
worker, 0.793
clerk, 0.756
cashier, 0.737
wench, 0.732
pharmacist, 0.724
person, 0.717
attendant, 0.702
saleswoman, 0.699
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: cashier
register, 0.867
counter, 0.809
clerk, 0.777
barista, 0.74
employee, 0.737
trainee, 0.732
sneered, 0.728
gent, 0.726

staffer, 0.721
greeter, 0.72
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: driver
dispatcher, 0.767
drivers, 0.7
operator, 0.694
shuttle, 0.687
bellman, 0.677
delivery, 0.674
mover, 0.671
dispatch, 0.67
estimator, 0.666
lyft, 0.665
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: saleswoman
staffer, 0.773
associate, 0.771
wench, 0.768
himi, 0.768
bartenderwaitress, 0.76
kash, 0.758
clerk, 0.757
doormen, 0.757
marissa, 0.756
saleslady, 0.753
--- --- --- --- --- --- --- ---
NEAREST NEIGHBORS FOR: salesman
salesperson, 0.812
dealership, 0.731
rep, 0.719
dealer, 0.718
mechanic, 0.717
sales, 0.711
salesmen, 0.696
robert, 0.69
finance, 0.69
mike, 0.684
--- --- --- --- --- --- --- ---