Digital Humanities First Assignment:

In [3]:
```
import os
import math
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("./data/feeding-america.csv", index_col = 'date')
df.head()
```

Out[3]:

date	book_id	ethnicgroup	recipe_class	region	ingredients
1922	fofb.xml	mexican	soups	ethnic	chicken;green pepper;rice;salt;water
1922	fofb.xml	mexican	meatfishgame	ethnic	chicken;rice
1922	fofb.xml	mexican	soups	ethnic	allspice;milk
1922	fofb.xml	mexican	fruitvegbeans	ethnic	breadcrumb;cheese;green pepper;pepper;salt;sar...
1922	fofb.xml	mexican	eggscheesedairy	ethnic	butter;egg;green pepper;onion;parsley;pepper;s...

In [4]:
```
print(len(df))        #Number of receipes
```

```
48032
```

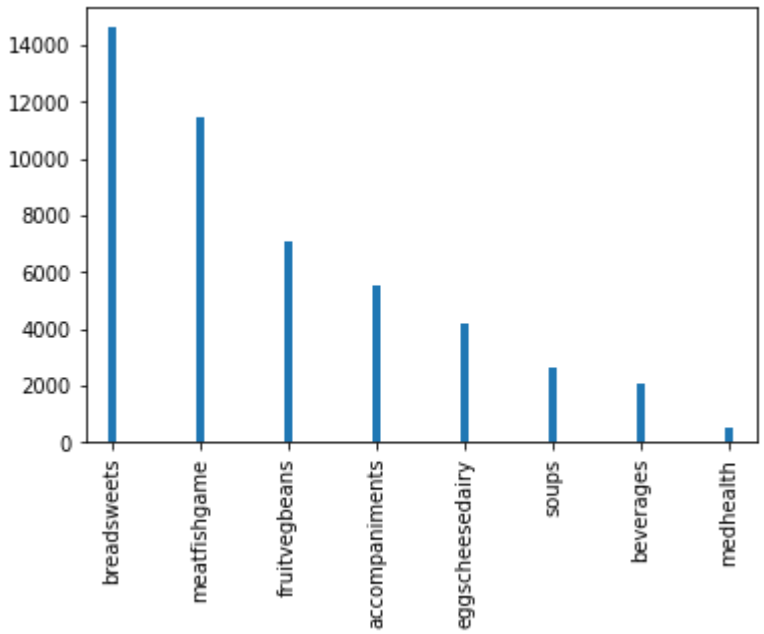
In [5]:
```
print(df['recipe_class'].unique())
```

```
['soups' 'meatfishgame' 'fruitvegbeans' 'eggscheesedairy' 'breadsweets'
 'beverages' 'accompaniments' 'medhealth']
```

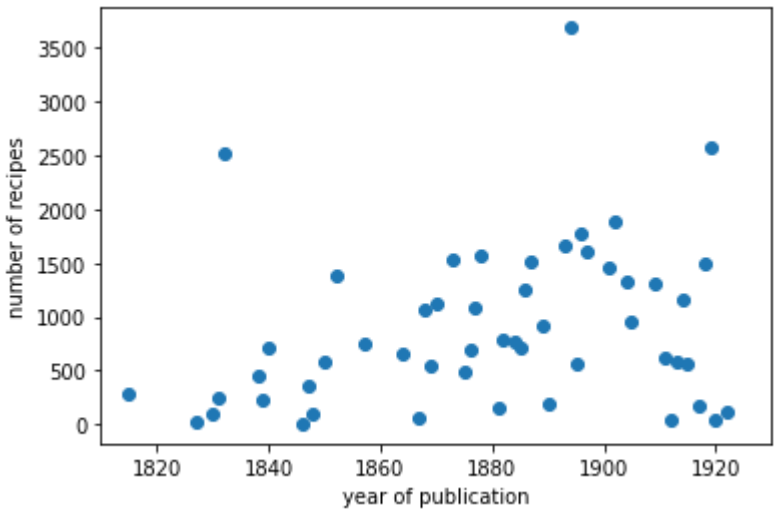
In [6]:
```
df['recipe_class'].value_counts()
```

Out[6]:
```
breadsweets        14630
meatfishgame       11477
fruitvegbeans       7085
accompaniments      5495
eggscheesedairy     4150
soups               2631
beverages           2031
medhealth            533
Name: recipe_class, dtype: int64
```

In [7]:
```
df['recipe_class'].value_counts().plot(kind='bar', color="C0", width=0.1)
plt.show()
```



In [8]:
```
grouped = df.groupby('date')
recipe_counts = grouped.size()
recipe_counts.plot(style='o', xlim=(1810, 1930))
plt.ylabel("number of recipes")
plt.xlabel("year of publication")
plt.show()
```



In [9]:
```
ingredients = df['ingredients'].str.split(';')
# group all rows from the same year
groups = ingredients.groupby('date')
# merge the lists from the same year
ingredients = groups.sum()
# compute counts per year
ingredients = ingredients.apply(pd.Series.value_counts).fillna(0)
# normalise the counts
ingredients = ingredients.divide(recipe_counts, 0)

edited_rows = ingredients.loc[((ingredients.index >= 1900) & (ingredients.index <
edited_rows
```

Out[9]:

date	butter	salt	water	flour	nutmeg	pepper	sugar	lemon	mace	
1827	0.000000	0.066667	0.600000	0.000000	0.033333	0.033333	0.400000	0.200000	0.000000	0
1830	0.234043	0.414894	0.489362	0.297872	0.127660	0.148936	0.329787	0.063830	0.021277	0
1831	0.467213	0.250000	0.352459	0.372951	0.122951	0.114754	0.471311	0.036885	0.028689	0
1832	0.377389	0.330414	0.382166	0.265525	0.125000	0.183121	0.296178	0.083599	0.086385	0
1838	0.466368	0.446188	0.448430	0.282511	0.141256	0.253363	0.159193	0.047085	0.082960	0
1839	0.413793	0.413793	0.461207	0.306034	0.120690	0.181034	0.219828	0.034483	0.073276	0
1840	0.433148	0.374652	0.576602	0.295265	0.224234	0.189415	0.271588	0.094708	0.147632	0
1901	0.348662	0.459163	0.464653	0.260810	0.037062	0.206589	0.446122	0.112560	0.010295	0
1902	0.410420	0.459862	0.440723	0.251462	0.064859	0.348751	0.219564	0.061138	0.028708	0
1904	0.397271	0.514784	0.426839	0.247157	0.043215	0.215315	0.335102	0.063685	0.029568	0
1905	0.439834	0.476141	0.495851	0.336100	0.069502	0.170124	0.483402	0.076763	0.019710	0
1909	0.398623	0.431523	0.437643	0.317521	0.055088	0.182096	0.405509	0.042846	0.014537	0
1911	0.473941	0.526059	0.418567	0.275244	0.061889	0.263844	0.332248	0.029316	0.017915	0
1912	0.358974	0.564103	0.666667	0.256410	0.051282	0.102564	0.230769	0.000000	0.025641	0
1913	0.422609	0.627826	0.495652	0.189565	0.041739	0.464348	0.300870	0.177391	0.055652	0
1914	0.382705	0.464897	0.345890	0.297945	0.032534	0.222603	0.351027	0.065925	0.006849	0
1915	0.545617	0.495528	0.513417	0.563506	0.073345	0.177102	0.574240	0.084079	0.007156	0
1917	0.005814	0.017442	0.151163	0.000000	0.139535	0.005814	0.348837	0.063953	0.000000	0
1918	0.297872	0.474734	0.486037	0.262633	0.036569	0.210771	0.375000	0.080452	0.010638	0
1919	0.393786	0.451262	0.240000	0.159612	0.017864	0.307961	0.128155	0.102524	0.003883	0
1920	0.000000	0.352941	0.676471	0.205882	0.000000	0.029412	0.000000	0.000000	0.000000	0

21 rows × 3532 columns

In [10]:
```
from sklearn.feature_selection import chi2

# Transform the index into a list of labels, in which each label
# indicates whether a row stems from before or after the Civil War:
labels = ['Early18th' if year < 1840 else 'Post19th' for year in edited_rows.inde

# replace missing values with zero (.fillna(0)),
# and compute the chi2 statistic:
keyness, _ = chi2(edited_rows.fillna(0), labels)

# Turn keyness values into a Series, and sort in descending order:
keyness = pd.Series(keyness, index=edited_rows.columns).sort_values(ascending=Fal
```

In [11]:
```
keyness.head(20)
```

Out[11]:
```
pearlash            0.495223
loaf sugar          0.449094
rice water          0.397945
wine                0.385085
baking powder       0.370681
vanilla             0.321912
molass              0.321107
beer                0.317949
yeast               0.314298
currant             0.282046
indian meal         0.271841
lemon peel          0.252384
gravy               0.217980
saltpetre           0.212098
lemon brandy        0.208674
ice                 0.199448
tomato              0.194492
soda                0.191709
granulated sugar    0.176807
salt                0.165199
dtype: float64
```

In [12]:
```
agg1 = edited_rows.loc[edited_rows.index > 1900].sum().rank(method='dense', pct=T
print(agg1.sort_values(ascending=False).head(10))
```

```
salt        1.000000
water       0.999167
butter      0.998333
sugar       0.997500
flour       0.996667
egg         0.995833
pepper      0.995000
milk        0.994167
egg yolk    0.993333
onion       0.992500
dtype: float64
```

In [13]:
```
agg2 = edited_rows.loc[ edited_rows.index < 1840 ].sum().rank(method='dense', pct
print(agg2.sort_values(ascending=False).head(10))
```

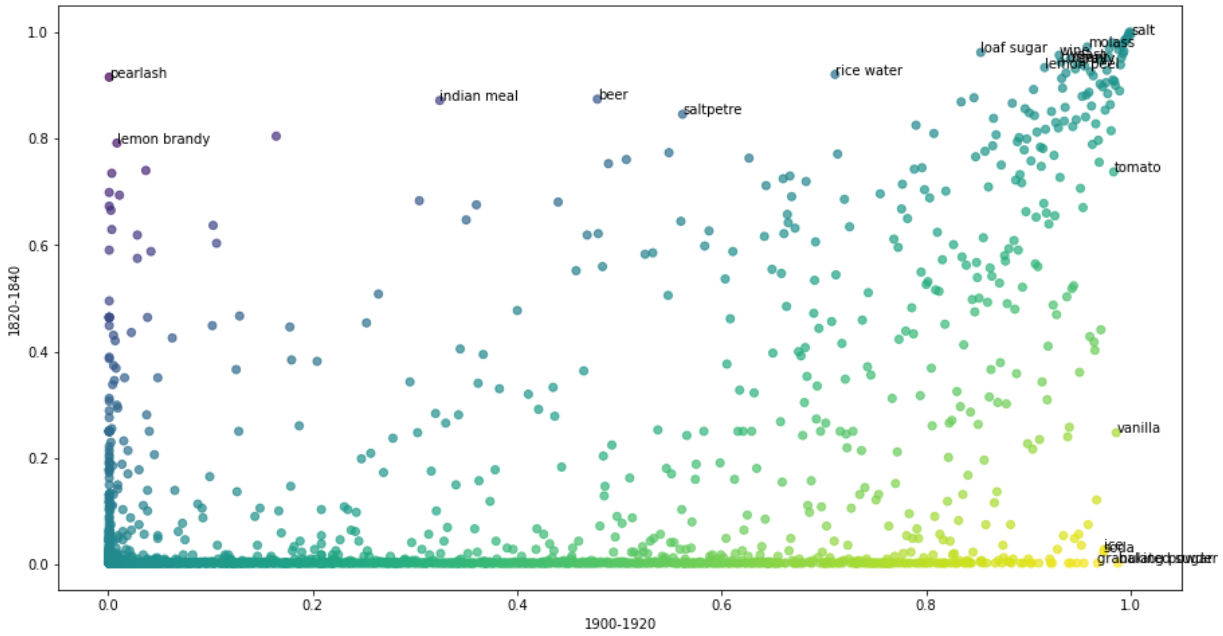
```
water      1.000000
butter     0.997423
salt       0.994845
sugar      0.992268
flour      0.989691
egg        0.987113
milk       0.984536
pepper     0.981959
nutmeg     0.979381
vinegar    0.976804
dtype: float64
```

In [14]:
```
rankings=pd.DataFrame({'1900-1920': agg1, '1820-1840': agg2})

fig = plt.figure(figsize=(15, 8))
plt.scatter(rankings['1900-1920'], rankings['1820-1840'],
            c=rankings['1900-1920'] - rankings['1820-1840'],alpha=0.7)
for i, row in rankings.loc[keyness.head(20).index].iterrows():
    plt.annotate(i, xy=(row['1900-1920'], row['1820-1840']))

plt.xlabel("1900-1920")
plt.ylabel("1820-1840")
```

Out[14]: Text(0, 0.5, '1820-1840')



END OF CODE!