Instacart and Online Ordering

Nahum Meherete
CMSC 678

The Problem

- Find out how users shop online
- Find trends that prove hypothesis of health and online food orders
- Create a Neural Network that does Market Basket Analysis

The Dataset

- The dataset is from 2017 and was released by Instacart
- The dataset contains over 3 million orders
- The dataset is split up into six tables
  - Orders, products, departments, aisles, order_products_train & order_products_prior

	product_id	product_name	aisle_id	department_id
0	1	Chocolate Sandwich Cookies	61	19
1	2	All-Seasons Salt	104	13
2	3	Robust Golden Unsweetened Oolong Tea	94	7
3	4	Smart Ones Classic Favorites Mini Rigatoni Wit...	38	1
4	5	Green Chile Anytime Sauce	5	13

Data Cleaning

In [8]:
```
big = pd.merge(aisles, products, on='aisle_id', how='inner')
```

In [9]:
```
big.head()
```

Out[9]:

	aisle_id	aisle	product_id	product_name	department_id
0	1	prepared soups salads	209	Italian Pasta Salad	20
1	1	prepared soups salads	554	Turkey Chili	20
2	1	prepared soups salads	886	Whole Grain Salad with Roasted Pecans & Mango ...	20
3	1	prepared soups salads	1600	Mediterranean Orzo Salad	20
4	1	prepared soups salads	2539	Original Potato Salad	20

In [10]:
```
big = pd.merge(big, departments, on='department_id', how='inner')
```

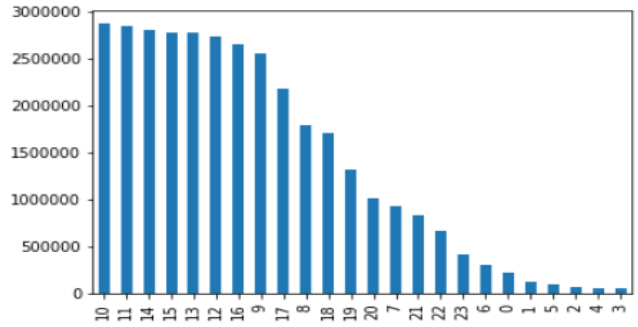
In [11]:
```
big.head()
```

Out[11]:

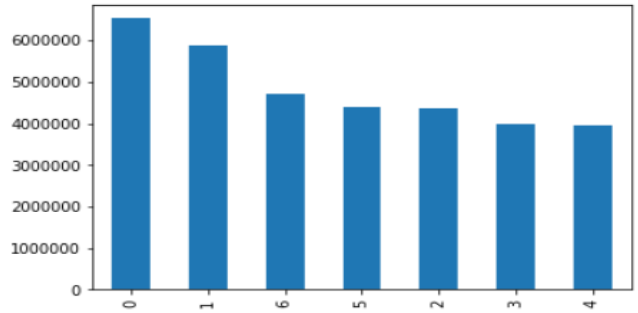
	aisle_id	aisle	product_id	product_name	department_id	department
0	1	prepared soups salads	209	Italian Pasta Salad	20	deli
1	1	prepared soups salads	554	Turkey Chili	20	deli
2	1	prepared soups salads	886	Whole Grain Salad with Roasted Pecans & Mango ...	20	deli
3	1	prepared soups salads	1600	Mediterranean Orzo Salad	20	deli
4	1	prepared soups salads	2539	Original Potato Salad	20	deli

Exploratory Analysis

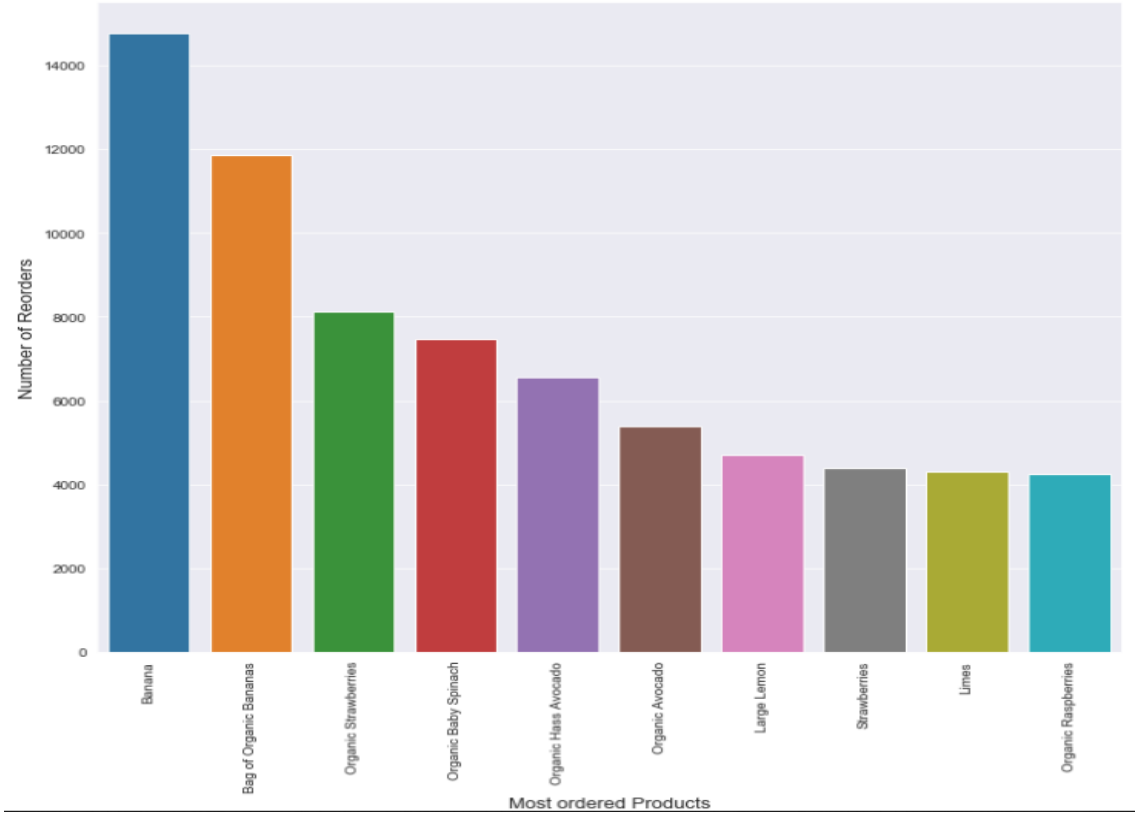
In [32]:
```
ax = big['order_hour_of_day'].value_counts().plot(kind='bar')
plt.show()
```



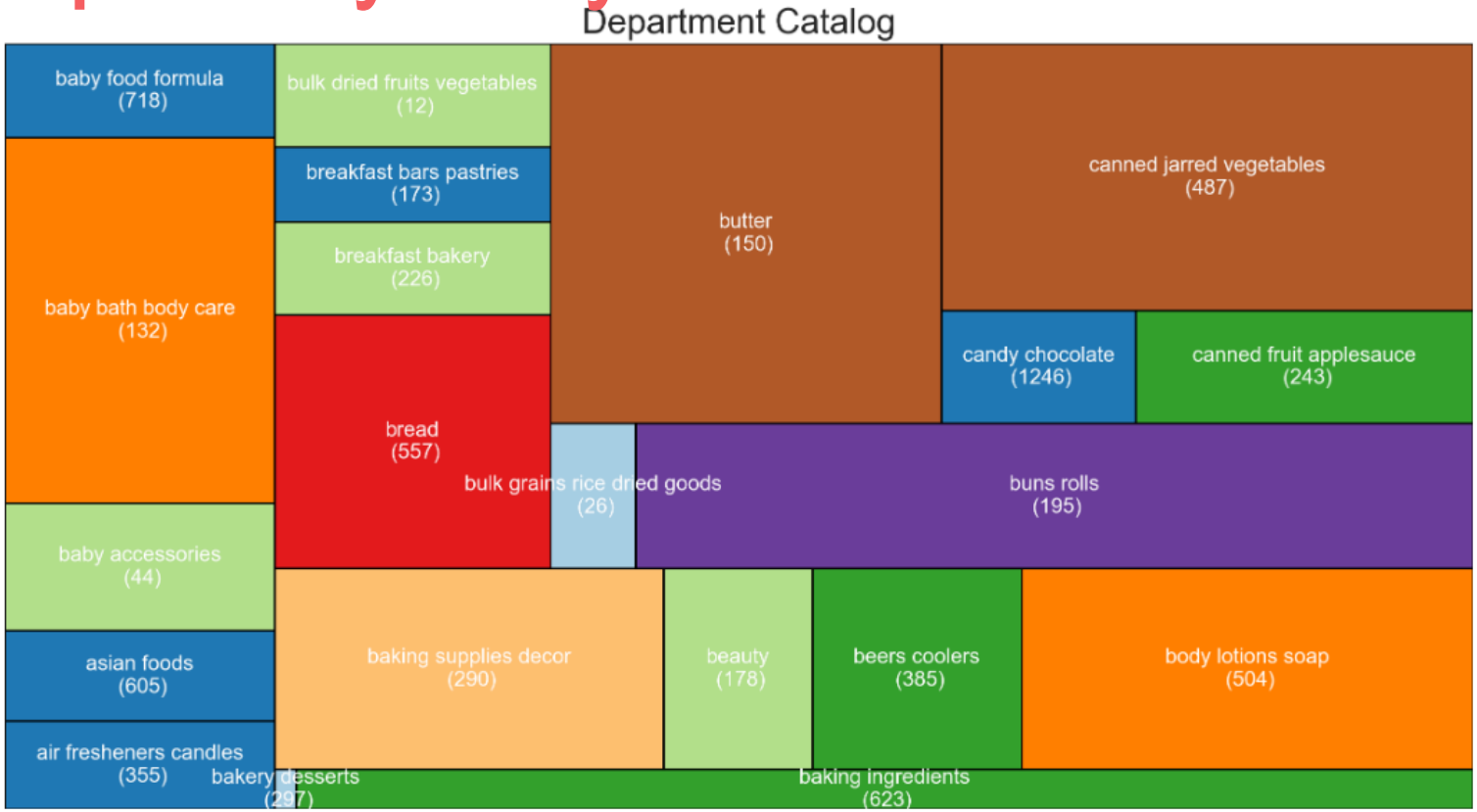
In [33]:
```
ax = big['order_dow'].value_counts().plot(kind='bar')
plt.show()
```



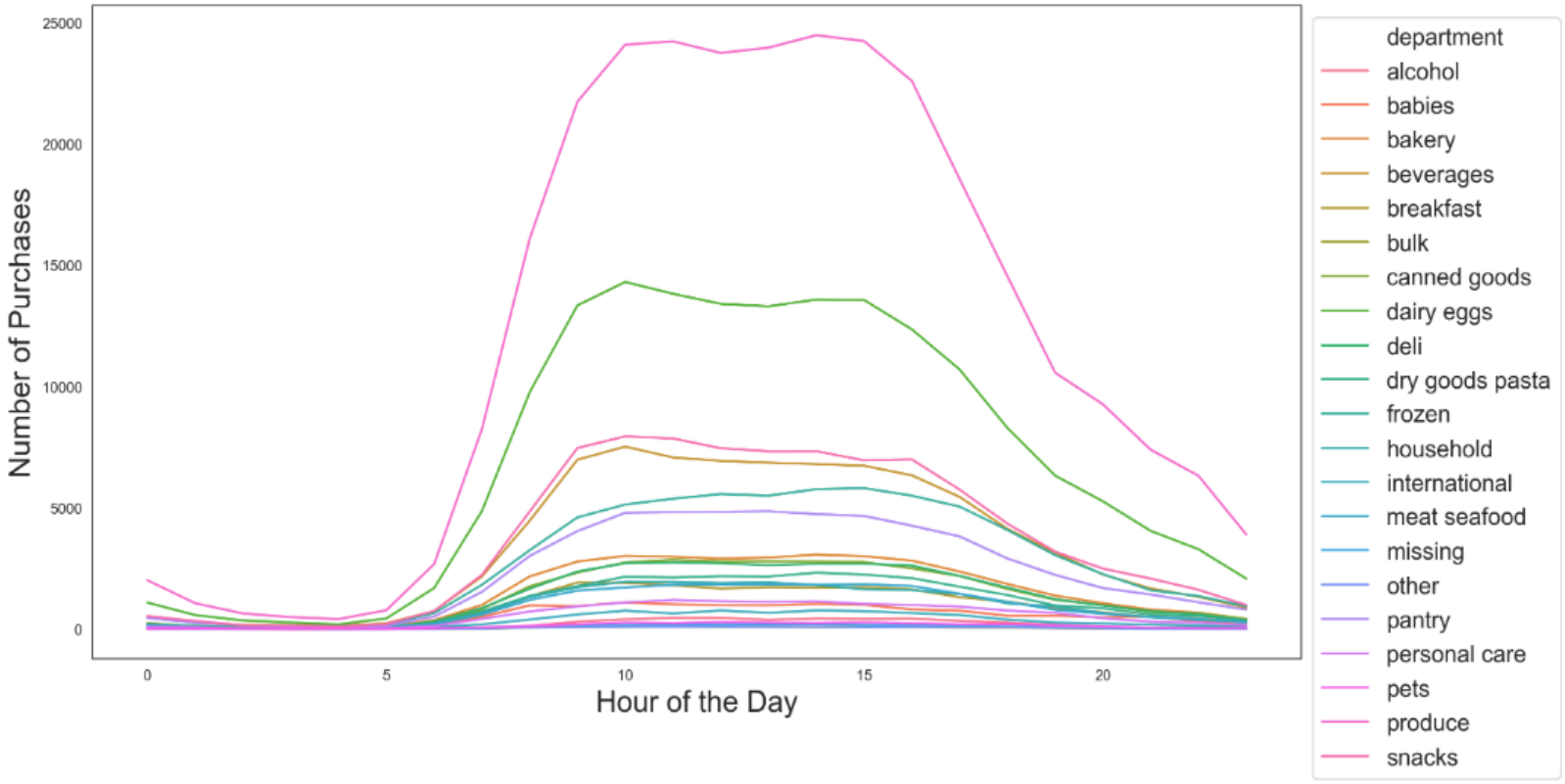
Exploratory Analysis Cont.



Exploratory Analysis Cont.



Exploratory Analysis Cont.



Neural Network Prep

```
big = pd.merge(X, cat, how='left', on='product_id')
big = big.iloc[0:1000000,]
```

```
y = big['reordered']
X = big.drop('reordered', axis=1)
encode = OneHotEncoder()
X = encode.fit_transform(X)
y = to_categorical(y)
```

```
svd = TruncatedSVD(n_components=100).fit(X)
X = svd.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
```

RandomSearchCV

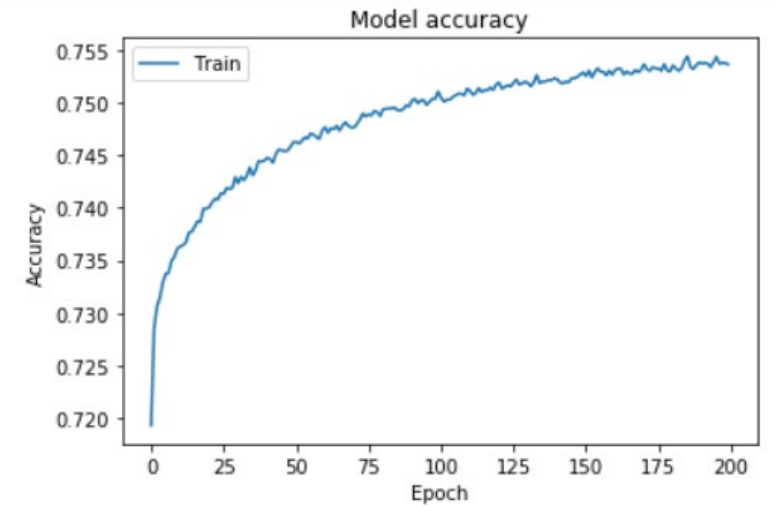
```
from keras.wrappers.scikit_learn import KerasClassifier

def create_model(optimizer='adam', neurons=64, dropout_rate=0.25):
    activation='relu'

    model = Sequential()
    model.add(BatchNormalization(axis=-1, momentum=.99, epsilon=.001, center=True, scale=True,
                                 beta_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros'))
    model.add(Dense(neurons, activation='relu'))
    model.add(Dropout(dropout_rate))
    model.add(Dense(neurons, activation='relu'))
    model.add(Dropout(dropout_rate))
    model.add(Dense(neurons, activation='relu'))
    model.add(Dropout(dropout_rate))
    model.add(Dense(2, activation='softmax'))

    model.compile(loss='categorical_crossentropy',
                  optimizer=optimizer,
                  metrics=['accuracy'])
    return model

model = KerasClassifier(build_fn=create_model, batch_size=128, epochs=50)
```

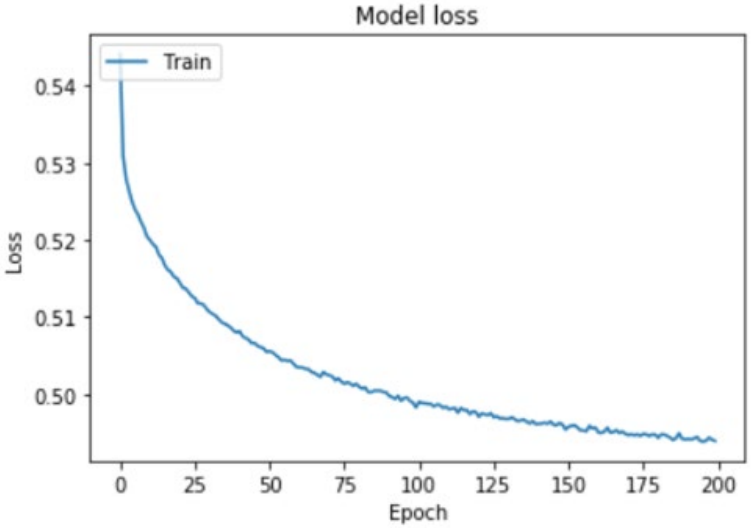
RandomSearchCV Results

```
from sklearn.model_selection import RandomizedSearchCV

params = {'neurons':[256,512],
          'dropout_rate':[0.25,0.5,0.75],
          'optimizer':['adam','sgd']}

grid = RandomizedSearchCV(estimator=model, param_distributions=params,
                          verbose=2, n_jobs=-1)
grid.fit(X_train, y_train)
grid.best_params_
```

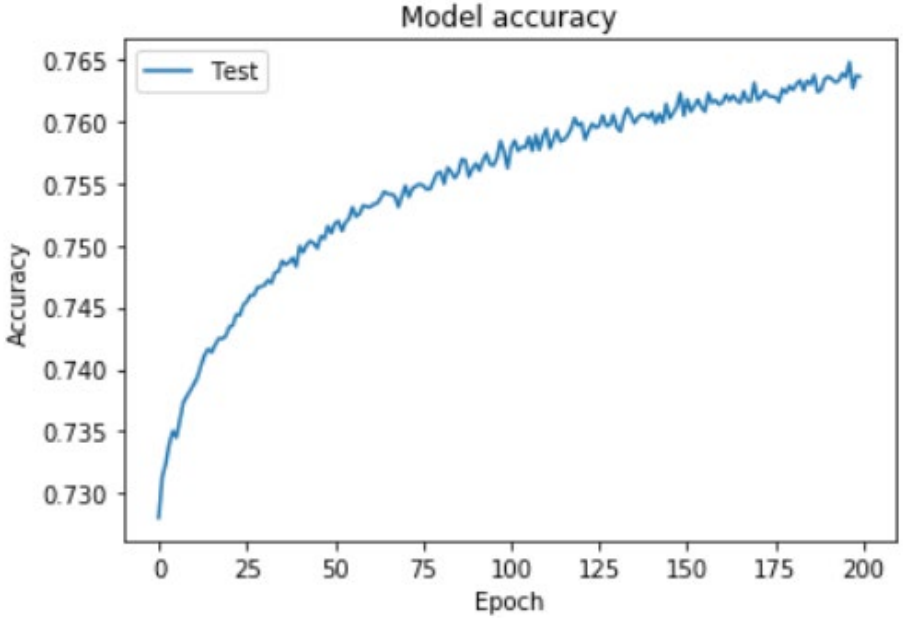
```
Epoch 42/50
670000/670000 [==============================] - 46s 69us/step - loss: 0.5300 - accuracy: 0.7293
Epoch 43/50
670000/670000 [==============================] - 46s 68us/step - loss: 0.5306 - accuracy: 0.7289
Epoch 44/50
670000/670000 [==============================] - 46s 69us/step - loss: 0.5304 - accuracy: 0.7292
Epoch 45/50
670000/670000 [==============================] - 46s 69us/step - loss: 0.5302 - accuracy: 0.7297
Epoch 46/50
670000/670000 [==============================] - 48s 72us/step - loss: 0.5300 - accuracy: 0.7298
Epoch 47/50
670000/670000 [==============================] - 46s 68us/step - loss: 0.5302 - accuracy: 0.7293
Epoch 48/50
670000/670000 [==============================] - 46s 68us/step - loss: 0.5304 - accuracy: 0.7294
Epoch 49/50
670000/670000 [==============================] - 45s 68us/step - loss: 0.5305 - accuracy: 0.7294
Epoch 50/50
670000/670000 [==============================] - 49s 74us/step - loss: 0.5302 - accuracy: 0.7296

{'optimizer': 'adam', 'neurons': 256, 'dropout_rate': 0.5}
```

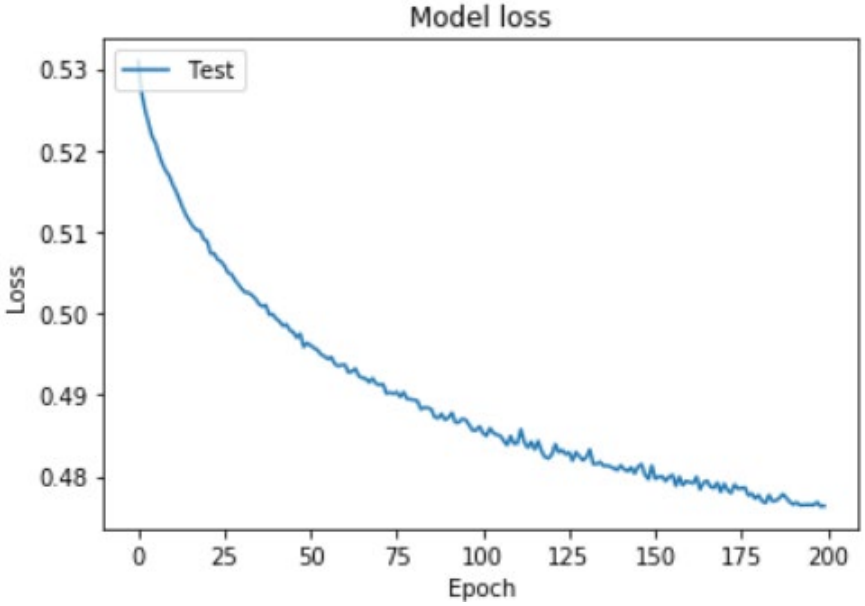
Training Data Results





Testing Data Results





References

Dang, A. K., Tran, B. X., Nguyen, C. T., Le, H. T., Do, H. T., Nguyen, H. D., . . . Ho, R. C. (2018). Consumer Preference and Attitude Regarding Online Food Products in Hanoi, Vietnam. *International Journal of Environmental Research and Public Health*. Retrieved September 28, 2019

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. **Scikit-learn: Machine Learning in Python**, Journal of Machine Learning Research, **12**, 2825-2830 (2011) (publisher link)

François Chollet, keras, (2015), GitHub repository, https://github.com/fchollet/keras

Fulton, A. (2012, July 23). *Ordering Food Online? That'll Be More Calories, Cost and Complexity*. Retrieved September 30, 2019, from NPR: https://www.npr.org/sections/thesalt/2012/07/23/157239715/ordering-food-online-thatll-be-more-calories-cost-and-complexity

John D. Hunter. **Matplotlib: A 2D Graphics Environment**, Computing in Science & Engineering, **9**, 90-95 (2007), DOI:10.1109/MCSE.2007.55 (publisher link)

Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. **The NumPy Array: A Structure for Efficient Numerical Computation**, Computing in Science & Engineering, **13**, 22-30 (2011), DOI:10.1109/MCSE.2011.37 (publisher link)

"The Instacart Online Grocery Shopping Dataset 2017", Accessed from https://www.instacart.com/datasets/grocery-shopping-2017 on 2019, September 09

Uri Laserson, squarify, (2019), GitHub repository, https://github.com/laserson/squarify

Wes McKinney. **Data Structures for Statistical Computing in Python**, Proceedings of the 9th Python in Science Conference, 51-56 (2010) (publisher link)