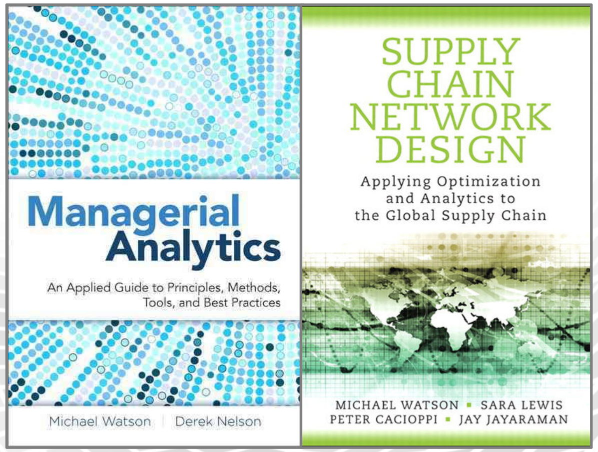
Welcome to the Webinar

Part II: Combining Optimization with Machine Learning for Better Decisions

Speaker Introduction

- Dr. Michael Watson
  - Managing Partner at Opex Analytics
  - Recognized leader in analytics and supply chain optimization
  - Michael was an early employee and leader at LogicTools
  - While at IBM, he was the worldwide business leader for network design, inventory and routing solutions
  - Co-author of ”Managerial Analytics” and “Supply Chain Network Design”
  - Adjunct professor at Northwestern University teaching graduate level courses within the Master in Engineering Management and the Master of Science in Analytics programs



Speaker Introduction

- Sam Hillis
  - Sr. Data Scientist at Opex Analytics
  - Areas of expertise include deep learning, sensor analytics, forecasting and big data
  - Earned a Bachelor of Science in Mathematics from the University of Illinois and a Master of Science in Analytics from Northwestern University
  - Sam's work at Opex has resulted in development of customized solutions predicting hourly changes in energy pricing, and frameworks for the real-time visualization and analysis of machine sensor data

Topics we plan to cover today

1	Machine Learning– management overview
2	Machine Learning – how it complements optimization
3	Machine Learning – learning through a case study
4	Useful tools and packages

Topics we plan to cover today

1	Machine Learning– management overview
2	Machine Learning – how it complements optimization
3	Machine Learning – learning through a case study
4	Useful tools and packages

Where Machine Learning Fits in Analytics

Descriptive
Reports
Drill Down
Visualization

Predictive
Machine Learning
Simulation
Forecasting
Alerts

Prescriptive
Optimization
Stochastic Opt

GUROBI OPTIMIZATION

opexanalytics

What is Possible with Machine Learning?

NETFLIX

amazon

Self Driving Cars

Why can machine learning algorithms do better

Linear Regression of 0/1 Response

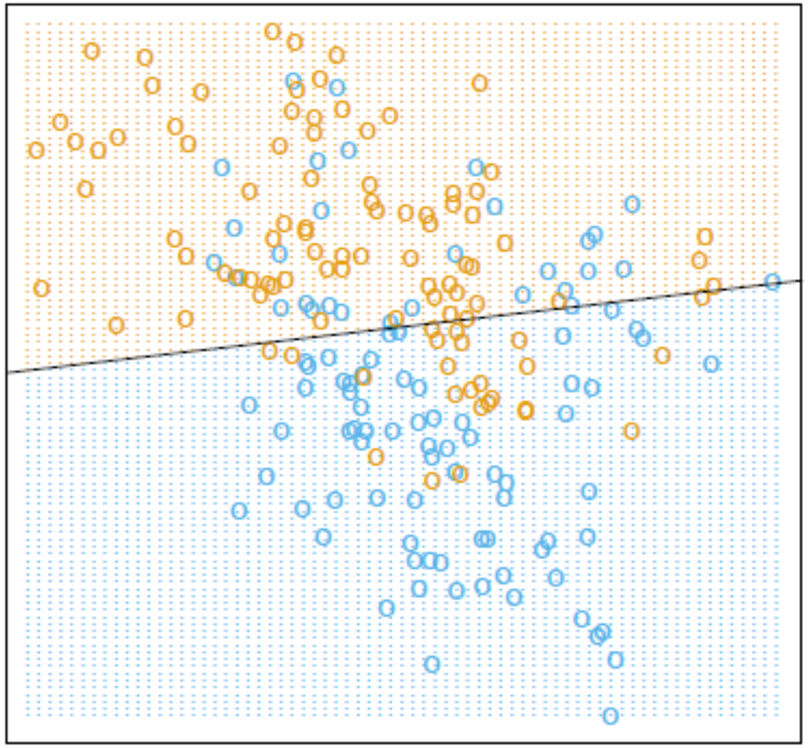


FIGURE 2.1. *A classification example in two dimensions. The classes are coded as a binary variable (*BLUE *= 0,* ORANGE *= 1), and then fit by linear regression. The line is the decision boundary defined by $x^T\hat{\beta} = 0.5$. The orange shaded region denotes that part of input space classified as* ORANGE*, while the blue region is classified as* BLUE.

Simple Regression

Source: https://statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf

Why can machine learning algorithms do better

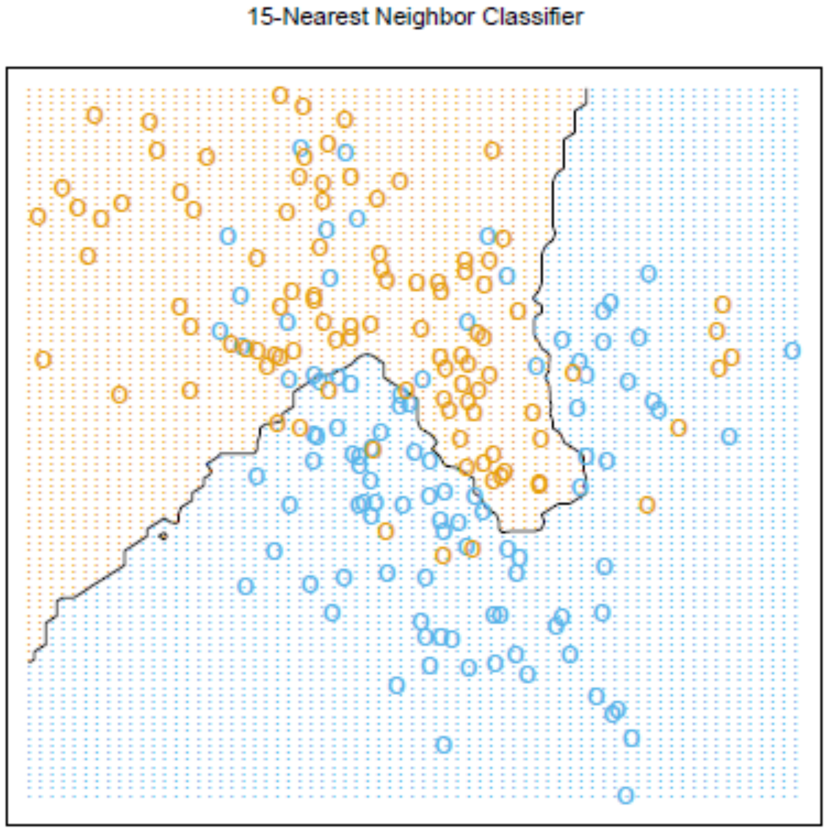


FIGURE 2.2. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$) *and then fit by 15-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the 15-nearest neighbors.*

Machine Learning can tease out patterns

Source: https://statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf

What do we mean by "Learning?"

How do I find bad potato chips?

Rules Based (not learning):

Build equations of all cases of what a bad chip looks like

Problem: Too many rules, takes too long, I still miss a lot; not flexible

Learning (as in machine learning)

Provide a database of pictures of chips with a label of "good" or "bad." The algorithm, not a person, figures out the rules– that is the learning.

How to classify Machine Learning problems

Supervised Learning

Classification
(non-quantitative data)

Regression
(quantitative data)

Unsupervised Learning

Clustering
(non-quantitative data)

Density Estimation
(quantitative data)

Supervised Learning
You have some labels for the data (you know what category it is in, you know what results it led to)

Unsupervised Learning
The data does not have labels on it

Learning
Taking a set of data and building a prediction model with it

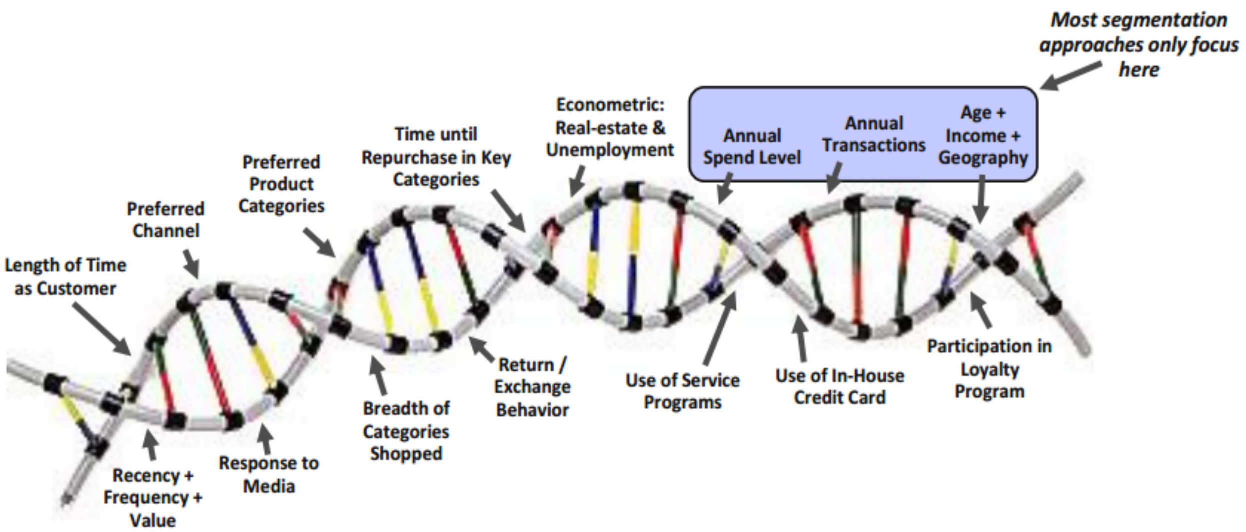

Unsupervised learning can uncover new insights
This is how we got “soccer moms”

Best Buy and Unsupervised learning

Stop Guessing: Analytics Can Tell You Who Your Customers Are

- Start with 30-40 modeled variables – "Feature Vectors"
- Each feature vector is like a gene, which describes a facet, or set of customer behavior traits
- 8-15 Feature Vectors are used to define Action Clusters, all 40 can be used to create sub-groups

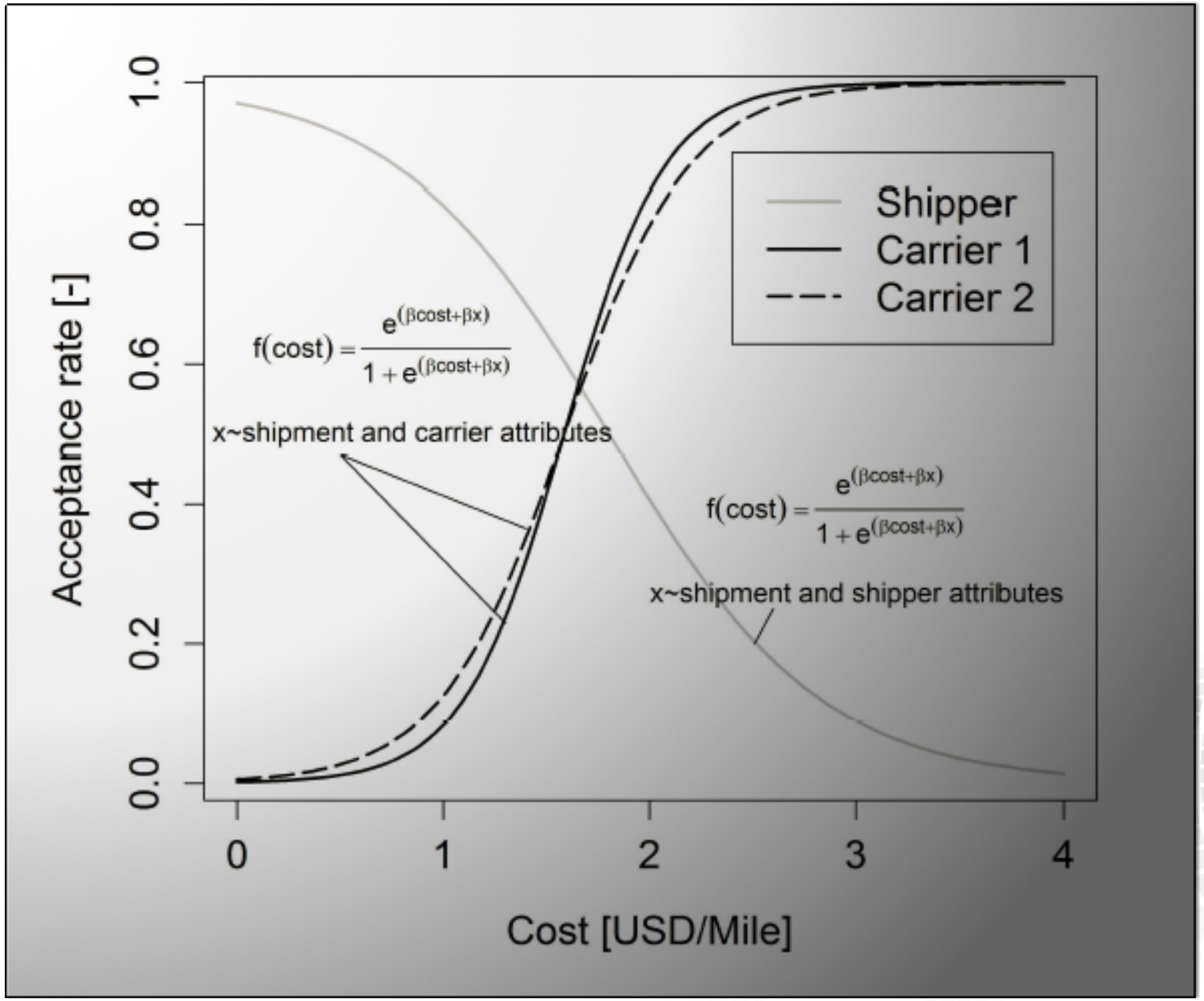
5-dimensional typical segmentation	≈ 16,000 views
12-dimensional Action Clusters	≈ 3,138,000,000,000 views
30 Feature Vectors	≈ 17,400,000,000,000,000,000,000,000,000,000 views



http://www.ibm.com/ibm/cioleadershipexchange/us/en/pdfs/Friesen.pdf

Predicting Market Sales with External Data and Feature Engineering

Predicting Prices in the trucking market



Source: http://opexanalytics.com/labs/pricing-logistics-services/

You can accelerate machine learning with more data

The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, *Google*

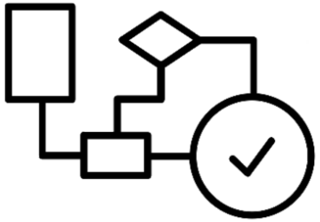
Using predictive models to find bad potato chips

1

Chips move at a high speed on the line

2

Pictures are taken



3

Pictures are classified based on K-NN algorithm

4

Bad" Chips are blown away

Source: http://www.econtalk.org/archives/2011/08/odonohoe_on_pot.html (at the 16:47 mark)

Topics we plan to cover today

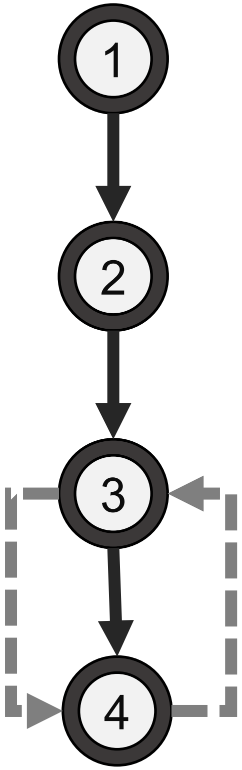
1 Machine Learning– management overview

2 Machine Learning – how it complements optimization

3 Machine Learning – learning through a case study

4 Useful tools and packages

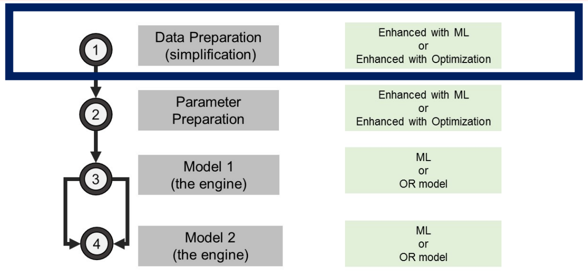
The typical modeling framework

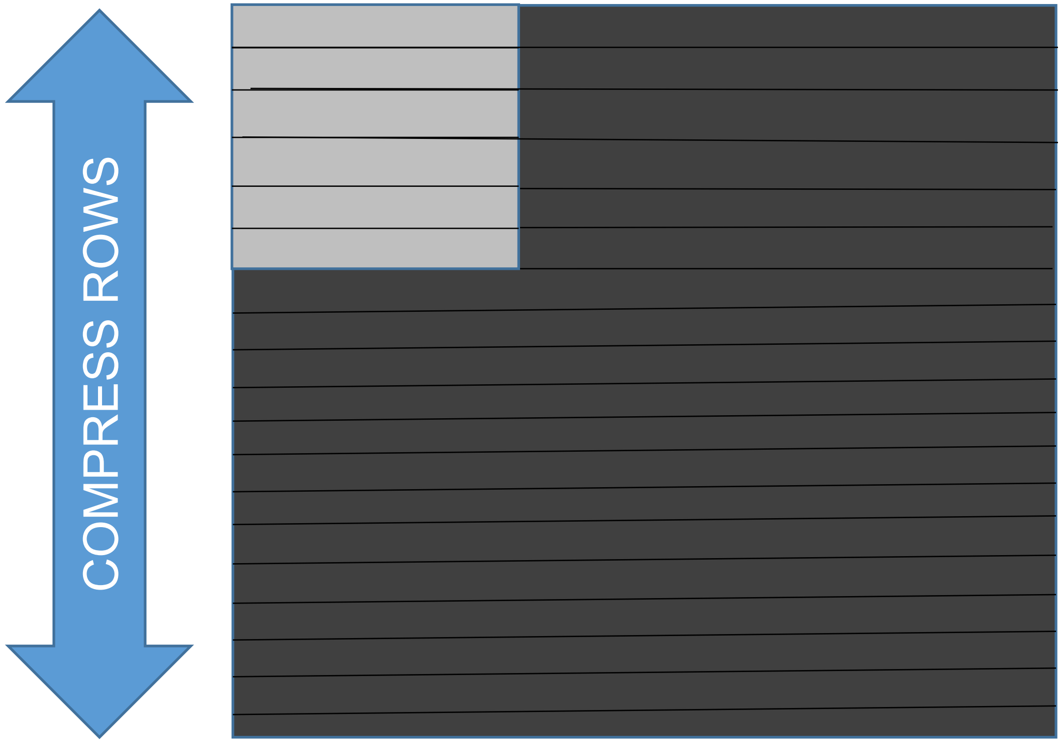


Data Preparation (simplification)	Enhanced with ML or Enhanced with Optimization
Parameter Preparation	Enhanced with ML or Enhanced with Optimization
Model 1 (the engine)	ML or OR model
Model 2 (the engine)	ML or OR model

Optimization experts may already be familiar with Multi Stage Optimization problems

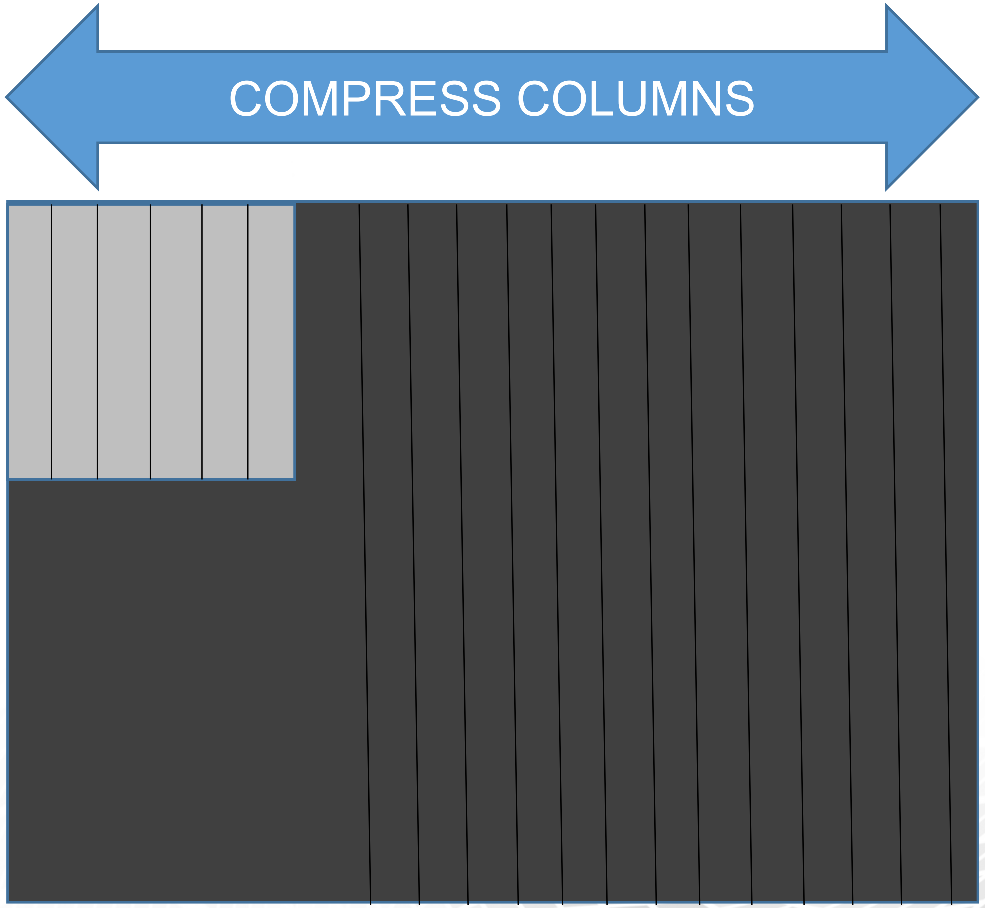
Machine Learning: Cut problem size





Clustering

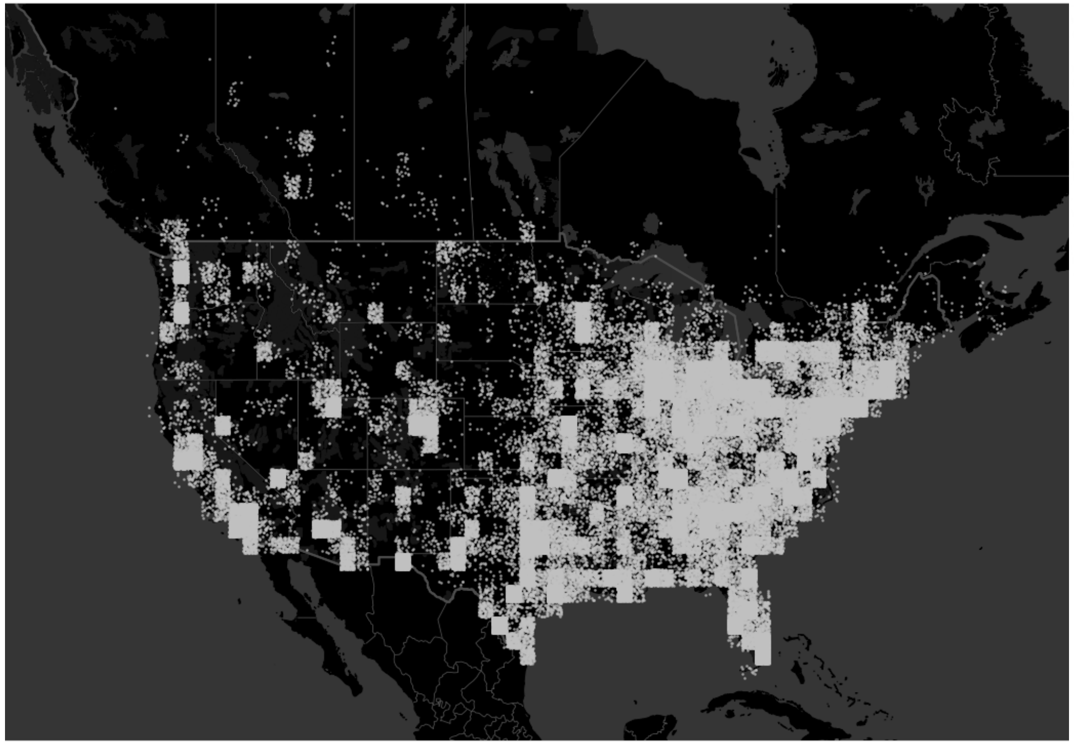
(K-means, hierarchical, etc.)

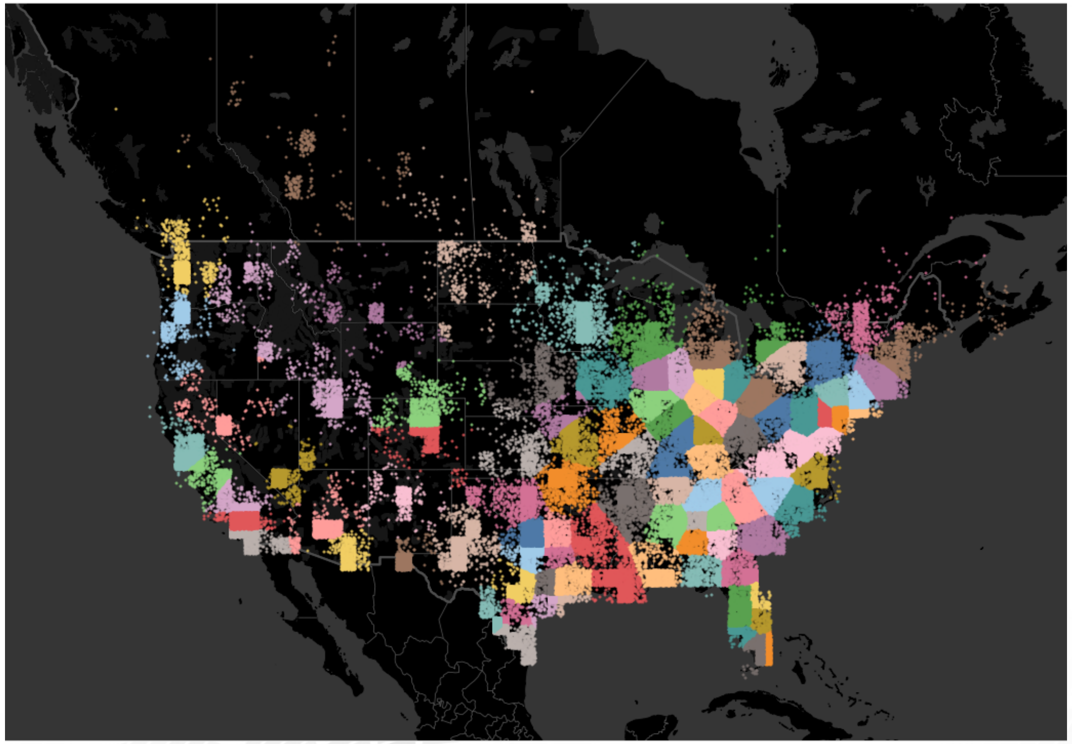


Factor Analysis

(Principal Component Analysis, etc.)

Machine Learning: Cut problem size (clustering)





Option 1: Optimization Model

Pros: Exact Answer

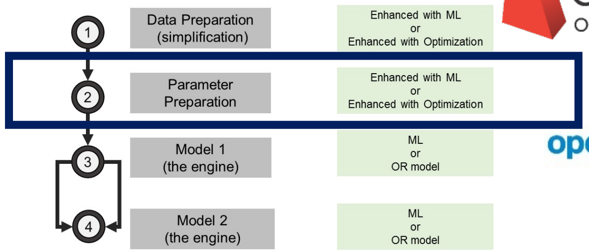
Cons: Single feature

Option 2: Unsupervised Learning Model

Pros: Fast, Multiple features

Cons: May not be ‘optimal’

Machine Learning: Automated Parameters



GUROBI OPTIMIZATION

opexanalytics

Demand by week

When does the season begin?

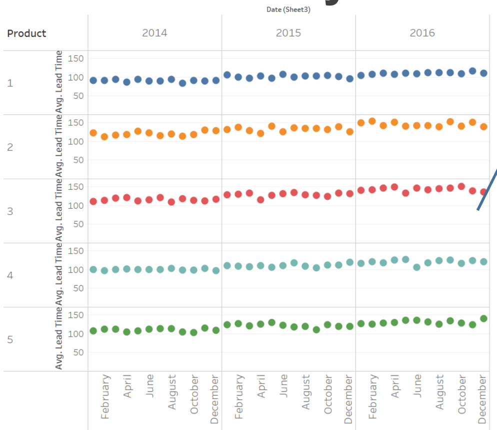
When does the season end?

What are the trends in lead time data?

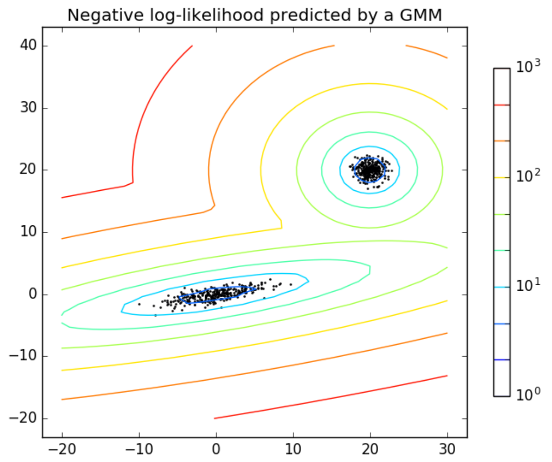
An online retailer has to worry about millions of products

Peak week Avg. Demand: 7500 Peak week Forecast Error: 1200 Lead Time: 94 days, std. dev = 11 days	Safety Stock: 2500 units
Peak week Avg. Demand: 6100 Peak week Forecast Error: 900 Lead Time: 105 days, std. dev = 19 days	Safety Stock: 1950 units

Lead Time by Product

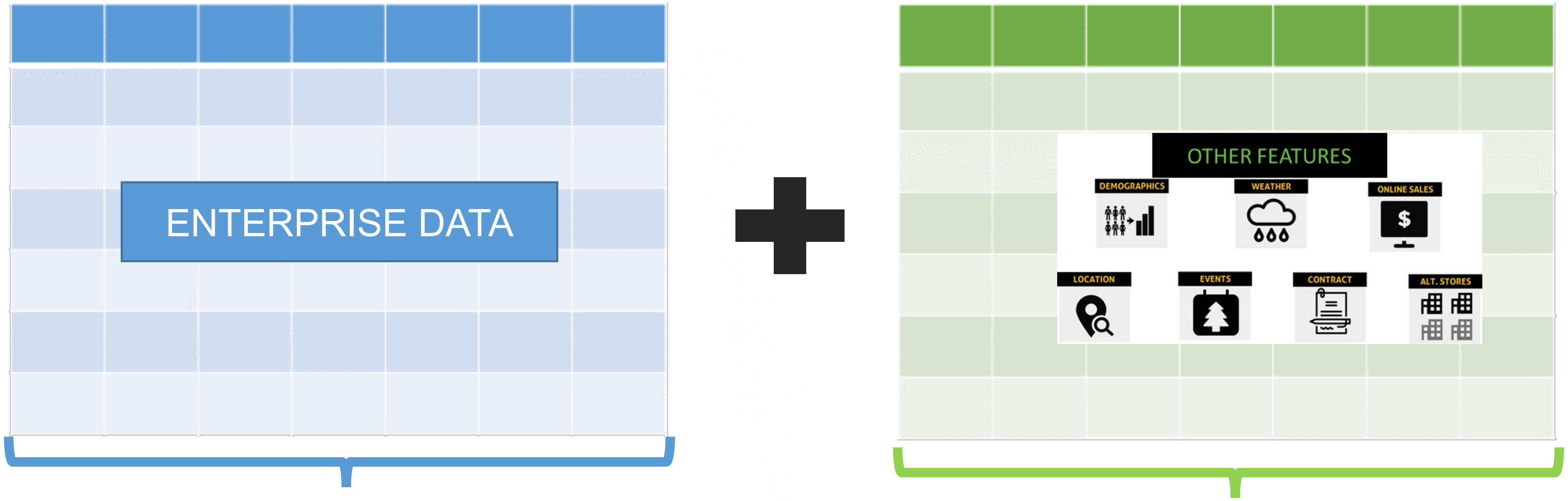


For the example above we have used Gaussian Mixture models to separate “Peak” from “Non Peak” months



Source: http://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_pdf.html

Machine Learning: Forecasting (to feed Optimization)



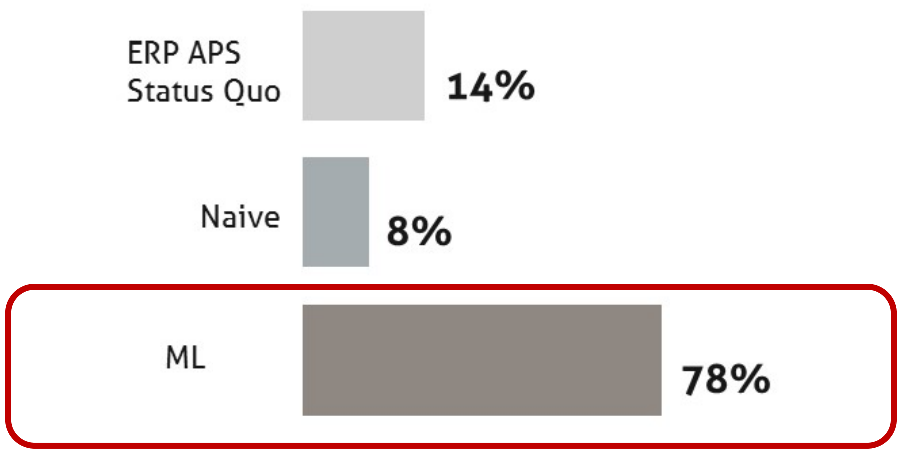
<u>Generally successful techniques</u>

1. Ensemble
2. Random Forest
3. SVM

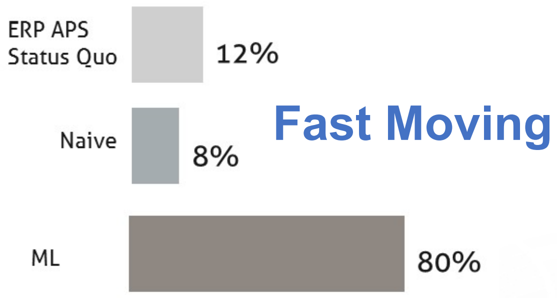
Machine Learning: Forecasting (to feed Optimization)

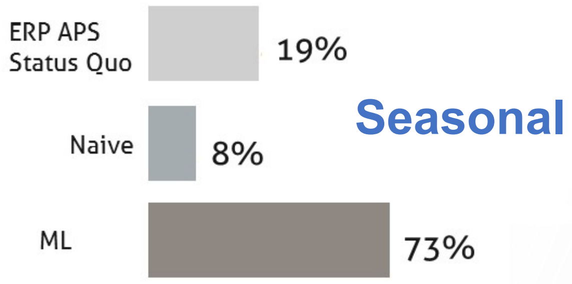
CPG CASE STUDY

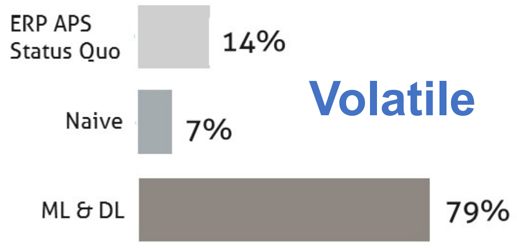
ML better than ERP APS Status Quo algorithms for 78% of the portfolio



Consistent results across portfolio at various stages of product life cycle







Topics we plan to cover today

1	Predictive Analytics – the basics
2	Machine Learning – how it complements optimization
3	Machine Learning – learning through a case study
4	Useful tools and packages

Case Study – Set up

Non asset based trucking company

Objective: Use past wins to predict competitive trucking rates on lanes nationwide in the US

- 1000s of lanes in the country
- Agents are responsible for quoting customers
- Historic data available only on booked lanes
- The predicted rates provides agents a guidance

Case Study – Some of the feature ‘families’

GUROBI OPTIMIZATION

opexanalytics

FLOW

Inflow/Outflow Ratio mattered

AGENT

Commission mattered

DISTANCE

CALENDAR

Did not matter

CUSTOMER

Did not matter

LOAD

RATE

Inflow/Outflow Ratio mattered

Case Study – Modeling Steps

0. Prepare data for modeling (distances, etc.)

1. Aggregate the locations to manageable ‘markets’

Approaches:
- Optimization
- Clustering

2. Calculate Features for Machine Learning

3. Dimensionality Reduction of Features

Approaches:
- Principal Component Analysis

4. Build & validate predictive model for baseline rates using machine learning

Approaches:
- Random Forest
- Ensemble

NON ASSET COMPANY

5. Built Network model to calibrate baseline rates for network imbalances

Approach:
- Network model (Linear Programming)

ASSET COMPANY

Case Study – Projection

Haversine calculation (“great circle distance”) is slow, use ECEF projection instead

```python
import numpy as np

# "borrowed" from https://stackoverflow.com/questions/10473852/convert-latitude-and-longitude-to-point-in-3d-space
def LLHtoECEF(lat, lon, alt):
  rad = 6378137.0 # Radius of the Earth (in meters)
  f = 1.0/298.257223563 # Flattening factor WGS84 Model
  cosLat = np.cos(lat)
  sinLat = np.sin(lat)
  FF = (1.0-f)**2
  C = 1/np.sqrt(cosLat**2 + FF * sinLat**2)
  S = C * FF
  x = (rad * C + alt)*cosLat * np.cos(lon)
  y = (rad * C + alt)*cosLat * np.sin(lon)
  z = (rad * S + alt)*sinLat
  return (x, y, z)
```

Case Study – Load Data

Load data

```
import pandas as pd
cluster_df = pd.read_csv('sample/cluster_input.csv')
cluster_df.head()
```

	FB_NUMBER	DATE_ID	DLAT	DLON
0	0	627	33.325080	-87.917297
1	3	627	40.738153	-84.030005
2	6	627	27.477675	-99.634112
3	11	627	38.863257	-85.634347
4	17	627	41.640130	-82.619150

Case Study – Apply Projections

Apply conversion; take note of how *zip* and *apply* are used

```
# perform ECEF conversion; don't forget to convert degrees to radians!!
cluster_df['x'], cluster_df['y'], cluster_df['z'] = zip(*cluster_df.apply(
    lambda x: LLHtoECEF(np.radians(x['DLAT']), np.radians(x['DLON']), 0.0), axis=1))

cluster_df.head()
```

	FB_NUMBER	DATE_ID	DLAT	DLON	x	y	z
0	0	627	33.325080	-87.917297	193875.702983	-5331228.727822	3484140.028965
1	3	627	40.738153	-84.030005	503357.190838	-4813370.318073	4140433.892498
2	6	627	27.477675	-99.634112	-947678.740954	-5582797.415327	2925275.736678
3	11	627	38.863257	-85.634347	378541.136700	-4958438.142641	3980508.233052
4	17	627	41.640130	-82.619150	613242.986358	-4734097.380054	4215816.035410

Case Study – Clustering

Load KMeans modeling object and create model

```
# load clustering function
from sklearn.cluster import KMeans
# create model object with 100 centroids
model = KMeans(n_clusters=100, n_init=10, n_jobs=-1)
```

Case Study – Clustering (continued)

Fit model to x-y-z coordinates

```
# fit model to ECEF coordinates
model.fit(cluster_df[['x','y','z']])
# add cluster to df
cluster_df['cluster'] = model.predict(cluster_df[['x','y','z']])
cluster_df.head()
```

	FB_NUMBER	DATE_ID	DLAT	DLON	x	y	z	cluster
0	0	627	33.325080	-87.917297	193875.702983	-5331228.727822	3484140.028965	94
1	3	627	40.738153	-84.030005	503357.190838	-4813370.318073	4140433.892498	43
2	6	627	27.477675	-99.634112	-947678.740954	-5582797.415327	2925275.736678	18
3	11	627	38.863257	-85.634347	378541.136700	-4958438.142641	3980508.233052	83
4	17	627	41.640130	-82.619150	613242.986358	-4734097.380054	4215816.035410	43

Case Study – (Alternate) MIP Approach

$$Minimize \quad \sum_i \sum_j (dist_{ij} * d_j * Y_{ij})$$ *Minimize weighted(by demand) average distance*

$$subject \ \ to$$

$$\sum_i Xi = P \quad \forall i$$ *Maximum of P open facilities*

$$\sum_i Yij = 1 \quad \forall j$$ *Ensure every customer is served (and by one source)*

$$Yij <= Xi \quad \forall i,j$$ *Customer can be served by a facility i only if facility is open*

$$Yij \in 0,1 \quad \forall i,j$$

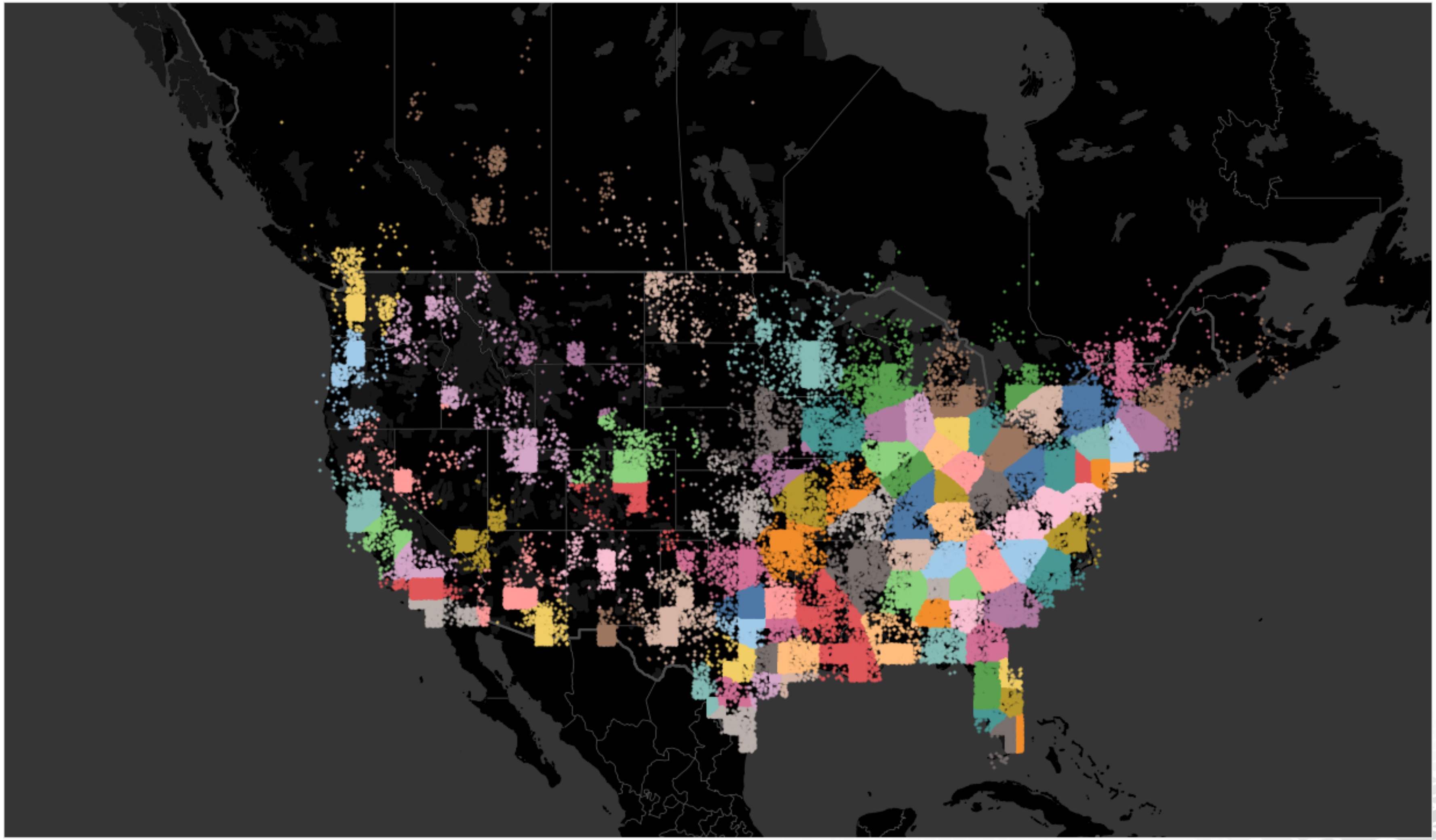
$$Xi \in 0,1 \quad \forall i$$

Xi =1 if facility open, Yij= 1 if customer j is served by facility i

Source: book, Supply Chain Network Design

```
m = Model()

#number of required centers
n = 100

#binary variable for open/close
x = m.addVars(l1, vtype = GRB.BINARY, name = 'x')

#binary variable for assigning location to an open center
y = m.addVars(l1, l1, vtype = GRB.BINARY, name = 'y')

#adding constraints
m.addConstrs((y[i,j] <= x[i] for i in l1 for j in l1), name = "cons_assign_to_open")
m.addConstrs((quicksum(y[i,j] for i in l1) == 1 for j in l1), name = "cons_one_source")
m.addConstr((quicksum(x[i] for i in l1) == n), name = "required_centers")
m.setObjective(quicksum(distance_data[i,j]*y[i,j]*demand_data.loc[j]['demand'] for i in l1 for j in l1) )
m.update()

#optimize
m.optimize()
```

Case Study – Segmentation Results



Case Study – Feature Engineering

Create feature tracking DataFrame

```
# master dataframe for tracking features
feat_master = cluster_df[['FB_NUMBER']].reset_index(drop=True)
feat_master.head()
```

	FB_NUMBER
0	0
1	3
2	6
3	11
4	17

Case Study – Feature Engineering (continued)

To get a count of the number of loads ending in a cluster in the last 15 days, begin by merging dataset on itself

```
# subset to just fb #'s, dates, and clusters
feat_df = cluster_df[['FB_NUMBER','DATE_ID','cluster']]
# merge dataframe on itself by cluster
feat_df = feat_df.merge(feat_df.rename(columns={'FB_NUMBER':'FB_NUMBER2','DATE_ID':'DATE_ID2'}), on='cluster')
feat_df.head()
```

	FB_NUMBER	DATE_ID	cluster	FB_NUMBER2	DATE_ID2
0	0	627	94	0	627
1	0	627	94	3700	626
2	0	627	94	3750	626
3	0	627	94	4218	627
4	0	627	94	4356	627

Case Study – Feature Engineering (continued)

Filter to get relevant date range

```
# subset to 15 days before
feat_df = feat_df[(feat_df['DATE_ID']>feat_df['DATE_ID2']) & (feat_df['DATE_ID']-feat_df['DATE_ID2']<=15)]
feat_df.head()
```

	FB_NUMBER	DATE_ID	cluster	FB_NUMBER2	DATE_ID2
1	0	627	94	3700	626
2	0	627	94	3750	626
3325	4218	627	94	3700	626
3326	4218	627	94	3750	626
4433	4356	627	94	3700	626

Case Study – Feature Engineering (continued)

Group by freight bill and get total count

```
# aggregate by fb and count loads ending in the same cluster in the last 15 days
feat_df = feat_df[['FB_NUMBER','FB_NUMBER2']].groupby('FB_NUMBER', as_index=False).count()
feat_df.rename(columns={'FB_NUMBER2':'LOADS_END15'}, inplace=True)
feat_df.head()
```

	FB_NUMBER	LOADS_END15
0	0	1
1	6	1
2	11	1
3	25	1
4	26	1

Case Study – Feature Engineering (continued)

Group by freight bill and get total count

```
# add feature back to master dataframe and fill NA's with 0's
feat_master = feat_master.merge(feat_df, how='left').fillna(0)
feat_master.head()
```

	FB_NUMBER	LOADS_END15
0	0	1
1	3	0
2	6	1
3	11	1
4	17	0

Case Study – Split Data

Split up data for model building

```
# sample feature file
feats = pd.read_csv('sample/features.csv').fillna(0)
# split train/validation/test
train = feats[feats['DATE_ID']<=feats['DATE_ID'].max()-40]
validation = feats[(feats['DATE_ID']<=feats['DATE_ID'].max()-20) & \
                    (feats['DATE_ID']>feats['DATE_ID'].max()-40)]
test = feats[feats['DATE_ID']>feats['DATE_ID'].max()-20]
```

Case Study – Dimensionality Reduction

Perform dimensionality reduction on dataset

```
# import principal components analysis library
from sklearn.decomposition import PCA
# create model object
model = PCA(n_components=0.99)
# fit model to training data
model.fit(train[train.columns[3:]])
# apply transformation to train/validation/test
X_tr = model.transform(train[train.columns[3:]])
X_va = model.transform(validation[train.columns[3:]])
X_te = model.transform(test[train.columns[3:]])
# check out the transformation
print(train.head())
print(X_tr[:5,:])
```

Case Study – Dimensionality Reduction

Original Dataset

```
   FB_NUMBER  DATE_ID      RATE  FEATURE_1  FEATURE_2  FEATURE_3  FEATURE_4  \
0          0      627  2.071805   2.065139   2.389623   2.368176         28
1          3      627  2.559242   2.509390   2.327381   2.259932         37
2          6      627  1.399007   1.339911   1.391344   1.390797        681
3         11      627  1.671642   1.671642   1.928783   1.988271        242
4         17      627  2.020408   2.020408   1.923893   1.912562        144

   FEATURE_5  FEATURE_6   FEATURE_7     ...      FEATURE_62  FEATURE_63  \
0        493  84.044000   77.662464     ...             719           1
1        211  66.076316   69.292025     ...            1359           1
2        604 109.031781  105.523317     ...            2516           1
3        335  49.821667   48.963140     ...             881           1
4        245  60.552632   59.006381     ...            1442           1

   FEATURE_64  FEATURE_65  FEATURE_66  FEATURE_67  FEATURE_68  FEATURE_69  \
0        1092           0    2.192891           0           1           0
1        1614           0    2.219208           0           1           0
2        1135           0    1.790774           0           1           0
3        1584           0    2.061148           0           1           0
4         889           0    2.095938           0           1           0

   FEATURE_70  FEATURE_71
0           6           0
1           4           0
2          37           0
3           6           0
4           7           0
```

Transformed Values

```
[[ 1301.8157762    794.46254941   410.74495319   -55.52077609
   -363.88676579]
 [  977.44797843  2391.57878914   159.64282395  -101.31166284
   -660.78018016]
 [-1410.9136752   2706.10335332 -2030.0697806   -731.86803437
    129.81522113]
 [  787.29685327  1727.55336435   737.47190343  -398.69307251
   -385.49179226]
 [ -135.60391357  1186.88862917  -871.95667574  -198.33688591
   -546.49488741]]
```

Case Study – Model Building and Validation

Create variables for model building

```
# load Random Forest regression model
from sklearn.ensemble import RandomForestRegressor
# range of values to test
n_trees = [10, 100]
n_feats = ['sqrt', 'log2']
max_depth = [7, 9, 11, 13]
# track scores
scores = []
```

Case Study – Model Building and Validation

Build model with all combinations of parameters

```
# grid search
for i in n_trees:
  for j in n_feats:
    for k in max_depth:
      print(i,j,k)
      # create model
      model = RandomForestRegressor(n_estimators=i,
                                    max_features=j,
                                    max_depth=k,
                                    n_jobs=-1)
      # fit to training data
      model.fit(X_tr, train['RATE'])
      # predict on validation setpredict(train[train.columns[3:]])
      preds = model.predict(X_va)
      # calculate pseudo-R^2
      r2 = 1 - ((validation['RATE']-preds)**2).sum()/((validation['RATE']-validation['RATE'].mean())**2).sum()
      scores.append(dict(r2=r2, n_trees=i, n_feats=j, max_depth=k))
```

Case Study – Model Building and Validation

Check out R-squared for each model on the validation set

```
print(pd.DataFrame(scores))
```

```
    max_depth n_feats  n_trees        r2
0           7    sqrt       10  0.203400
1           9    sqrt       10  0.212975
2          11    sqrt       10  0.218592
3          13    sqrt       10  0.225572
4           7    log2       10  0.204371
5           9    log2       10  0.213731
6          11    log2       10  0.221816
7          13    log2       10  0.225626
8           7    sqrt      100  0.206476
9           9    sqrt      100  0.218043
10         11    sqrt      100  0.230300
11         13    sqrt      100  0.239658
12          7    log2      100  0.206313
13          9    log2      100  0.218773
14         11    log2      100  0.229910
15         13    log2      100  0.239538
```

Case Study – Model Building and Validation

Rebuild model with both training and validation using best set of parameters and run test through it

```
model = RandomForestRegressor(n_estimators=100,
                              max_features='sqrt',
                              max_depth=13,
                              n_jobs=-1)
# fit to training & validation data
model.fit(np.concatenate([X_tr, X_va]), np.concatenate([train['RATE'], validation['RATE']]))
# predict on test set
preds = model.predict(X_te)
# calculate pseudo-R^2
r2 = 1 - ((test['RATE']-preds)**2).sum()/((test['RATE']-test['RATE'].mean())**2).sum()

print(r2)
```

```
0.319998292826
```

Topics we plan to cover today

1	Machine Learning– management overview
2	Machine Learning – how it complements optimization
3	Machine Learning – learning through a case study
4	Useful tools and packages

Useful Tools and Packages: NumPy

Data Analysis

NumPy

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Data Visualization

matplotlib

Data Access

SQLAlchemy

Spark

Computation

scikit learn

SciPy

StatsModels
Statistics in Python

Deep Learning

TensorFlow™

theano

Useful Tools and Packages: Pandas

What is it?

Library adding support for DataFrames – column oriented data structures

Why is it important?

- DataFrames are an essential tool for data analysis in Python
- Toolkit for aggregating, transforming, and manipulating data indispensable for model preparation
- Easy to learn, great for non-programmers or spreadsheet-oriented analysts

Useful Tools and Packages: Sci-kit Learn

What is it?

Go-to package for machine learning algorithms

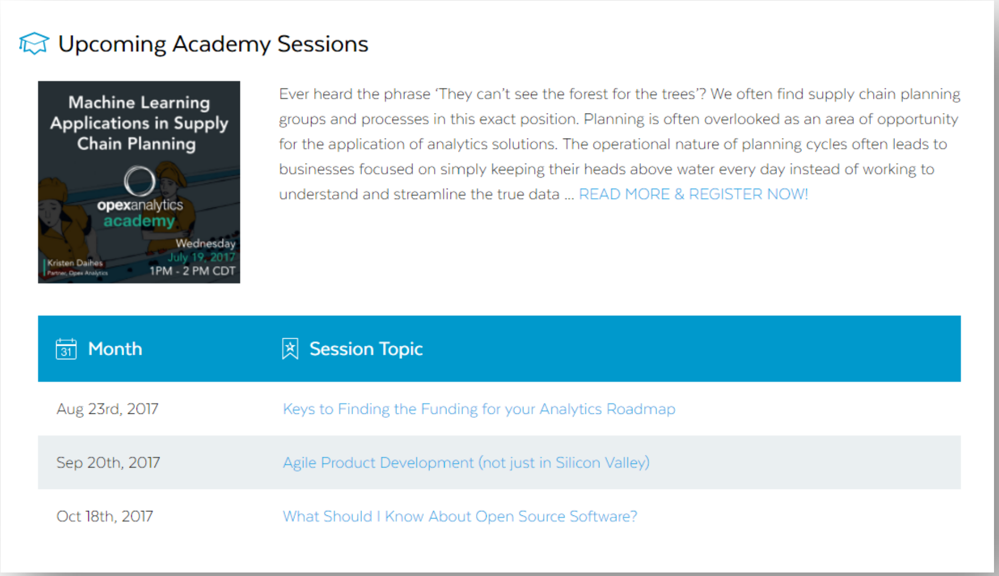
Why is it important?

- Contains massive set of tools for data analysis and learning including:
  - Cross-validation and model selection
  - Feature extraction and data preparation
  - Unsupervised learning (clustering, PCA, etc.)
  - Supervised learning (linear models, ensemble methods, svm, etc.)

Thank you for joining us

- If you haven't already done so, please register at http://www.gurobi.com
- Visit http://www.gurobi.com/downloads/get-anaconda to try Gurobi and Python for yourself.
- For questions about Gurobi pricing contact sales@gurobi.com or sales@gurobi.de.
- A recording of this webinar, including the slides, will be available in roughly one week.

For other Opex Analytics Academy led sessions check out http://www.opexanalytics.com/academy/



Upcoming Academy Sessions

Ever heard the phrase 'They can't see the forest for the trees'? We often find supply chain planning groups and processes in this exact position. Planning is often overlooked as an area of opportunity for the application of analytics solutions. The operational nature of planning cycles often leads to businesses focused on simply keeping their heads above water every day instead of working to understand and streamline the true data ... READ MORE & REGISTER NOW!

Month	Session Topic
Aug 23rd, 2017	Keys to Finding the Funding for your Analytics Roadmap
Sep 20th, 2017	Agile Product Development (not just in Silicon Valley)
Oct 18th, 2017	What Should I Know About Open Source Software?