## DATA SCIENCE

Exercise 8

| inputlookup bcg\_churn.csv

| sample partitions=3 seed=42

| where partition\_number < 2

| fit FieldSelector mode=k\_best param=5 type=categorical inactive from \*

| fields fs\_\*

L8S1

| inputlookup phishing.csv

| fit FieldSelector

type=categorical

mode=percentile

param=5

Result from \*

| table fs\_\*

## L8S2

| inputlookup phishing.csv

| sample partitions=2 seed=1234

| search partition\_number=0

| fit SVM Result from SSLfinal\_State into SVM\_classifier

## L8S3

L8S4

| inputlookup phishing.csv

| sample partitions=2 seed=123456

| search partition\_number=1

| apply SVM\_classifier as prediction

| `classificationstatistics(prediction, Request\_URL)`

## DS Module 8 - Slide 208

| inputlookup diabetes.csv

## DS Module 8 - Slide 213

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number < 2

| fit LogisticRegression response from BMI age into LogisticRegressionClassifier

## DS Module 8 - Slide 215

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number < 2

| fit LogisticRegression response from BMI age into LogisticRegressionClassifier probabilities=t

## DS Module 8 - Slide 216

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number = 2

| apply LogisticRegressionClassifier as prediction

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number < 2

| fit LogisticRegression response from BMI age into LogisticRegressionClassifier probabilities=t

## DS Module 8 - Slide 217

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number < 2

| fit SVM into svm\_model response from BMI age

## DS Module 8 - Slide 218

| inputlookup diabetes.csv

| sample partitions=3 seed=42

| search partition\_number = 2

| apply svm\_model as prediction

| `confusionmatrix(response, prediction)`

## L7S2

| inputlookup auto-mpg.csv

| sample partitions=2 seed=123456

| search partition\_number=0

| fit LinearRegression hp from displ cyl into hp

| `regressionstatistics(hp,predicted(hp))`

## L7S3

sourcetype=access\_combined earliest=-6w@w latest=-2w@w

| timechart count span=1d

| fit LinearRegression count from \_time as prediction into web

## L7S4

sourcetype=access\_combined earliest=-2w@w latest=+6w@w

| timechart count span=1d

| apply web

L7S5

sourcetype=access\_combined earliest=-6w@w latest=-2w@w

| timechart count span=1d

| eval t2 = \_time \* \_time

| fit LinearRegression count from \_time t2 into quad\_prediction

| fields - t2

## Geoff Market Segmentation Analysis - Outlier Detection and Removal

index=retail Country="United Kingdom" Quantity>0 UnitPrice>0 CustomerID=\*

| eval monetary = Quantity \* UnitPrice

| eval recency = now() - \_time

| eventstats min(recency) as minrecency

| eval recency = round((recency - minrecency)/86400)

| stats sum(monetary) as feature\_monetary, dc(InvoiceNo) as feature\_frequency, min(recency) as feature\_recency by CustomerID

| eventstats p99(feature\_\*) as p99\_\*

| where feature\_monetary < p99\_monetary AND feature\_frequency < p99\_frequency AND feature\_recency < p99\_recency

| fit StandardScaler feature\_\*

| fields CustomerID SS\_\*

| untable CustomerID feature value

| bin value span=1.0

| chart count by value feature

## Geoff Market Segmentation Analysis

index=retail Country="United Kingdom" Quantity>0 UnitPrice>0 CustomerID=\*

| eval monetary = Quantity \* UnitPrice

| eval recency = now() - \_time

| eventstats min(recency) as minrecency

| eval recency = round((recency - minrecency)/86400)

| stats sum(monetary) as feature\_monetary, dc(InvoiceNo) as feature\_frequency, min(recency) as feature\_recency by CustomerID

| eventstats p99(feature\_\*) as p99\_\*

| where feature\_monetary < p99\_monetary AND feature\_frequency < p99\_frequency AND feature\_recency < p99\_recency

| fit StandardScaler feature\_\*

| fit KMeans k=4 SS\_feature\_monetary SS\_feature\_frequency SS\_feature\_recency

| table cluster feature\_\*

## Geoff Market Segmentation Analysis - anomalydetection command

index=retail Country="United Kingdom" Quantity>0 UnitPrice>0 CustomerID=\*

| eval monetary = Quantity \* UnitPrice

| eval recency = now() - \_time

| eventstats min(recency) as minrecency

| eval recency = round((recency - minrecency)/86400)

| stats sum(monetary) as feature\_monetary, dc(InvoiceNo) as feature\_frequency, min(recency) as feature\_recency by CustomerID

| anomalydetection method=iqr action=transform feature\_monetary feature\_frequency feature\_recency

| fit StandardScaler feature\_\*

| fit KMeans k=4 SS\_feature\_monetary SS\_feature\_frequency SS\_feature\_recency

| where isnotnull(cluster)

| table cluster feature\_\*

## Geoff Transactional Analysis

sourcetype="access\_combined" action=\*

| reverse

| streamstats count as stage by JSESSIONID

| xyseries JSESSIONID stage action

| fillnull value=.

| stats count by 1 2 3 4 5

| sort - count

| foreach \*

[ eval <<FIELD>>=if('<<FIELD>>'=".", NULL, '<<FIELD>>')]

## \_L6S1

earliest=-1d index=main sourcetype=cisco\*

| cluster field=punct t=0.8 labelonly=t showcount=t match=ngramset

| stats values(\_raw) as raw max(\_time) as time by cluster\_label, cluster\_count

| eval hour = now() - (60 \* 60)

| where time > hour

| sort cluster\_count

| fields cluster\_count, cluster\_label, raw

## \_L6S2

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 avg(count) as avg, stdev(count) as stdev

| eval multiplier = 3

| eval lower\_bound = avg - (stdev \* multiplier)

| eval upper\_bound = avg + (stdev \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## \_L6S3

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 avg(count) as avg, stdev(count) as stdev

| eval multiplier = 2

| eval lower\_bound = avg - (stdev \* multiplier)

| eval upper\_bound = avg + (stdev \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## \_L6S4

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 median(count) as medianCount

| eval absDev=(abs('count'-medianCount))

| streamstats window=100 median(absDev) as medianAbsDev

| eval lowerBound=(medianCount-medianAbsDev\*15), upperBound=(medianCount+medianAbsDev\*15)

| eval outlier=if(count < lowerBound OR count > upperBound, 1, 0)

| table \_time count lowerBound upperBound outlier

## \_L6S5

| inputlookup portscans.csv

| timechart span=1d count

| eventstats median(count) as median, p25(count) as p25, p75(count) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lower\_bound = median - (IQR \* multiplier)

| eval upper\_bound = median + (IQR \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## \_L6S5

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time, host

| eventstats median(count) as median, p25(count) as p25, p75(count) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lowerBound = median - (IQR \* multiplier)

| eval upperBound = median + (IQR \* multiplier)

| eval outlier = if(count < lowerBound OR count > upperBound, 1, 0)

| table \_time count lowerBound upperBound outlier

## \_L6S6

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time, host

| eventstats median(count) as median, p25(count) as p25, p75(count) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lowerBound = median - (IQR \* multiplier)

| eval upperBound = median + (IQR \* multiplier)

| eval outlier = if(count < lowerBound OR count > upperBound, 1, 0)

| table \_time count lowerBound upperBound outlier

## \_L6S7

| inputlookup portscans.csv

| timechart span=1d dc(src) as distinctCount

| eventstats median(distinctCount) as median, p25(distinctCount) as p25, p75(distinctCount) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lower\_bound = median - (IQR \* multiplier)

| eval upper\_bound = median + (IQR \* multiplier)

| eval outlier = if(distinctCount < lower\_bound OR distinctCount > upper\_bound, 1, 0)

| table \_time distinctCount lower\_bound upper\_bound outlier

## \_L6S9

| inputlookup call-center.csv

## L6S1

index=main sourcetype=cisco\* earliest=-1d

| cluster field=punct t=0.8 labelonly=t showcount=t match=ngramset

| stats values(\_raw) as raw min(\_time) as time by cluster\_label, cluster\_count

| eval one\_hour\_ago = now() - (60 \* 60)

| where time > one\_hour\_ago

| sort -cluster\_count

| fields cluster\_count, cluster\_label, raw

## L6S2

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 avg(count) as avg, stdev(count) as stdev

| eval multiplier = 3

| eval lower\_bound = avg - (stdev \* multiplier)

| eval upper\_bound = avg + (stdev \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## L6S3

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 avg(count) as avg, stdev(count) as stdev

| eval multiplier = 2

| eval lower\_bound = avg - (stdev \* multiplier)

| eval upper\_bound = avg + (stdev \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## L6S4

## L6S4

| inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time

| streamstats window=100 median(count) as medianCount

| eval absDev=(abs('count'-medianCount))

| streamstats window=100 median(absDev) as medianAbsDev

| eval lowerBound=(medianCount-medianAbsDev\*15),

upperBound=(medianCount+medianAbsDev\*15)

| eval outlier=if(count < lowerBound OR count > upperBound, 1, 0)

| table \_time count lowerBound upperBound outlier

## L6S4

| inputlookup portscans.csv

| timechart span=1d count

| eventstats median(count) as median, p25(count) as p25, p75(count) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lower\_bound = median - (IQR \* multiplier)

| eval upper\_bound = median + (IQR \* multiplier)

| eval outlier = if(count < lower\_bound OR count > upper\_bound, 1, 0)

| table \_time count lower\_bound upper\_bound outlier

## L6S6

|inputlookup portscans.csv

| bin \_time span=1d

| stats count by \_time, host

| eventstats median(count) as median, p25(count) as p25, p75(count) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lowerBound = median - (IQR \* multiplier)

| eval upperBound = median + (IQR \* multiplier)

| eval outlier = if(count < lowerBound OR count > upperBound, 1, 0)

| table \_time count lowerBound upperBound outlier

## L6S7

| inputlookup portscans.csv

| timechart span=1d dc(src) as distinctCount

| eventstats median(distinctCount) as median, p25(distinctCount) as p25, p75(distinctCount) as p75

| eval IQR = p75 - p25

| eval multiplier = 2

| eval lower\_bound = median - (IQR \* multiplier)

| eval upper\_bound = median + (IQR \* multiplier)

| eval outlier = if(distinctCount < lower\_bound OR distinctCount > upper\_bound, 1, 0)

| table \_time distinctCount lower\_bound upper\_bound outlier