Microsoft

Real-time ML Journey

MicrosoftML + Revoscale

Hiram Fleitas

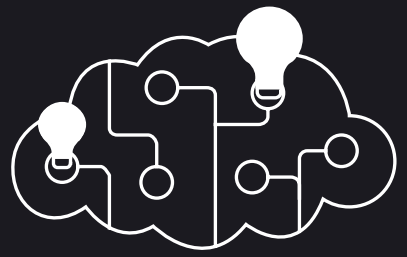
Senior Customer Engineer
Microsoft Data & AI

badges

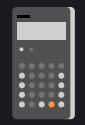
About Me

aka.ms/hiram

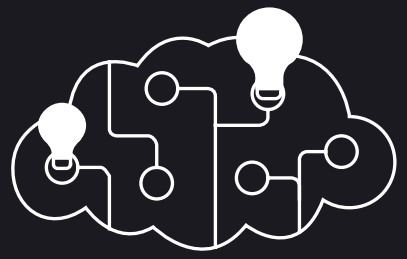
1. Machine Learning
2. Synapse Analytics
3. Power Platform
4. DevOps



1995 1999 2000s 2017 2021



Legal, Medical, Retail, Travel Insurance, ISV, HR, Cruise Line, PnC Insurance, plus more…



2017 2018 2019 2020 2021



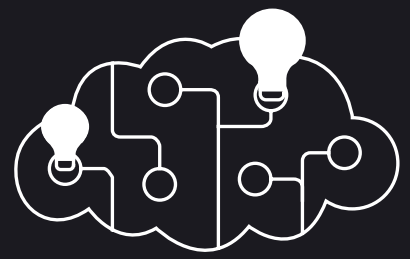










1 2 3 4 5 6 7 8

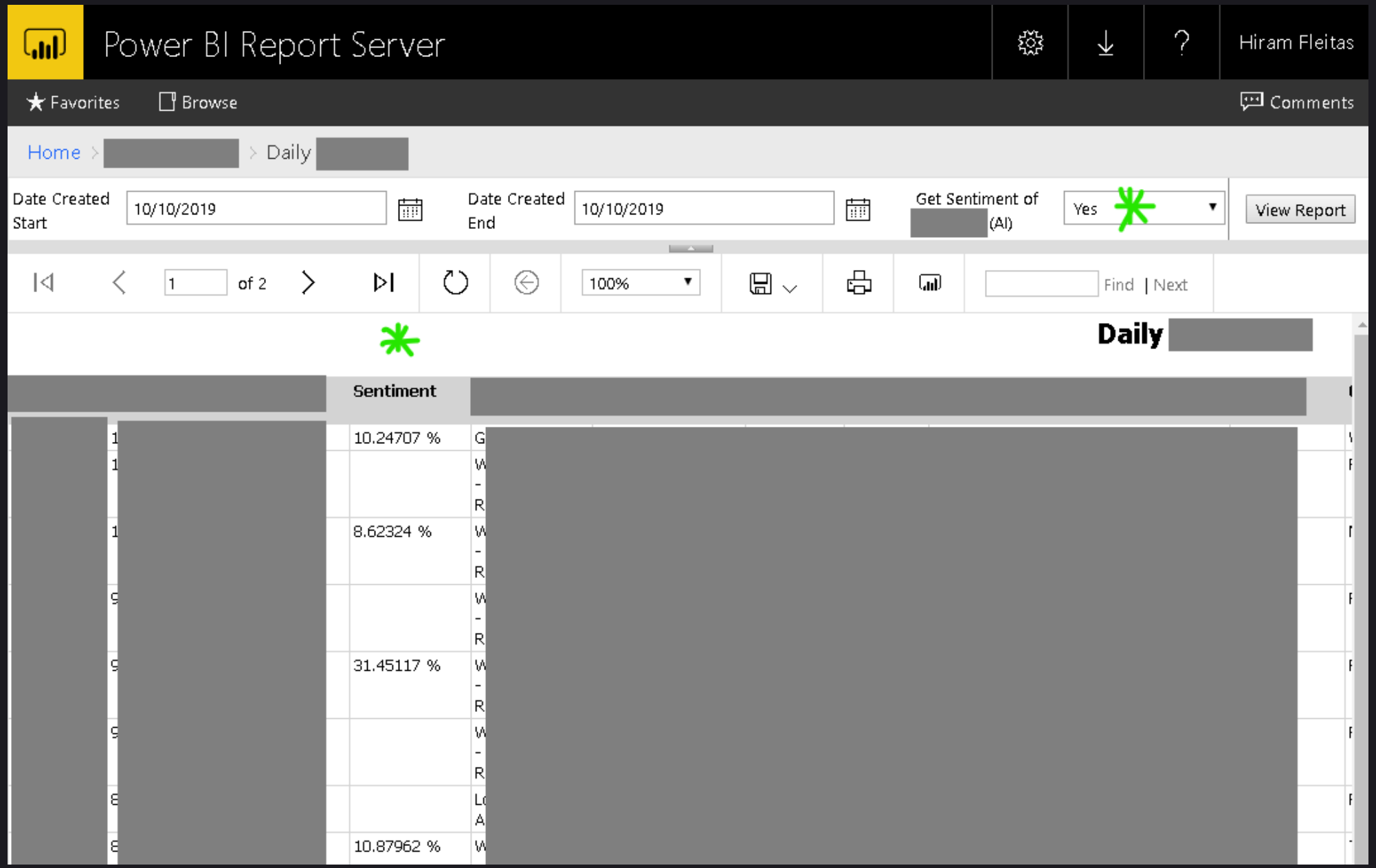
Iris Ski Text Forms Image LoS Price ∞

Agenda

with impact

1. Sentiment $30M
2. QnA Chat Bots $200K
3. Forms $2M
4. Capital Modeling $500M
5. Length of Stay $40M
6. Search by Photo $200K

Sentiment $30M



Learn more:
[Hide Column based up Parameter Selected SSRS]

```
create or alter proc GetCognitiveAPIQuoteSentiment
as
    set nocount on;
    declare @py nvarchar(max);

    set @py = N'import requests, pprint as pr
from pandas.io.json import json_normalize

apikey = "mykey"
api = "https://eastus2.api.cognitive.microsoft.com/text/analytics/v2.0/"
url = api + "sentiment"

df = jsondocs

headers  = {
     "Ocp-Apim-Subscription-Key": apikey,
     "content-type": "application/json"
}
response = requests.post(
     url,
     headers = headers,
     data = df.iloc[0][0].encode()
)

rds = response.json()
df2 = json_normalize(rds, "documents")

pr.pprint(rds)
print(type(df2),df2,sep="\n")
';
```

```
    drop table if exists apiresults;
    create table apiresults (id int, score float);

    insert into apiresults
    exec sp_execute_external_script @language = N'Python'
        ,@script = @py
        ,@input_data_1 = N'select * from JsonQuotes'
        ,@input_data_1_name = N'jsondocs'
        ,@output_data_1_name =  N'df2'
    select * from apiresults;

    update  q
        set q.Sentiment = a.Score
    from    Quotes q
    inner join apiresults a
        on q.quoteid = a.id
    where   q.Sentiment is null;
go
```

Learn more:
github.com/hfleitas/seattle2019/blob/master/2SentimentPrediction/Trollhunters.ipynb

Subject: Re: Developer API - Reviews Text Truncated

JUL 31, 2018 | 02:49PM PDT

Hi Hiram,

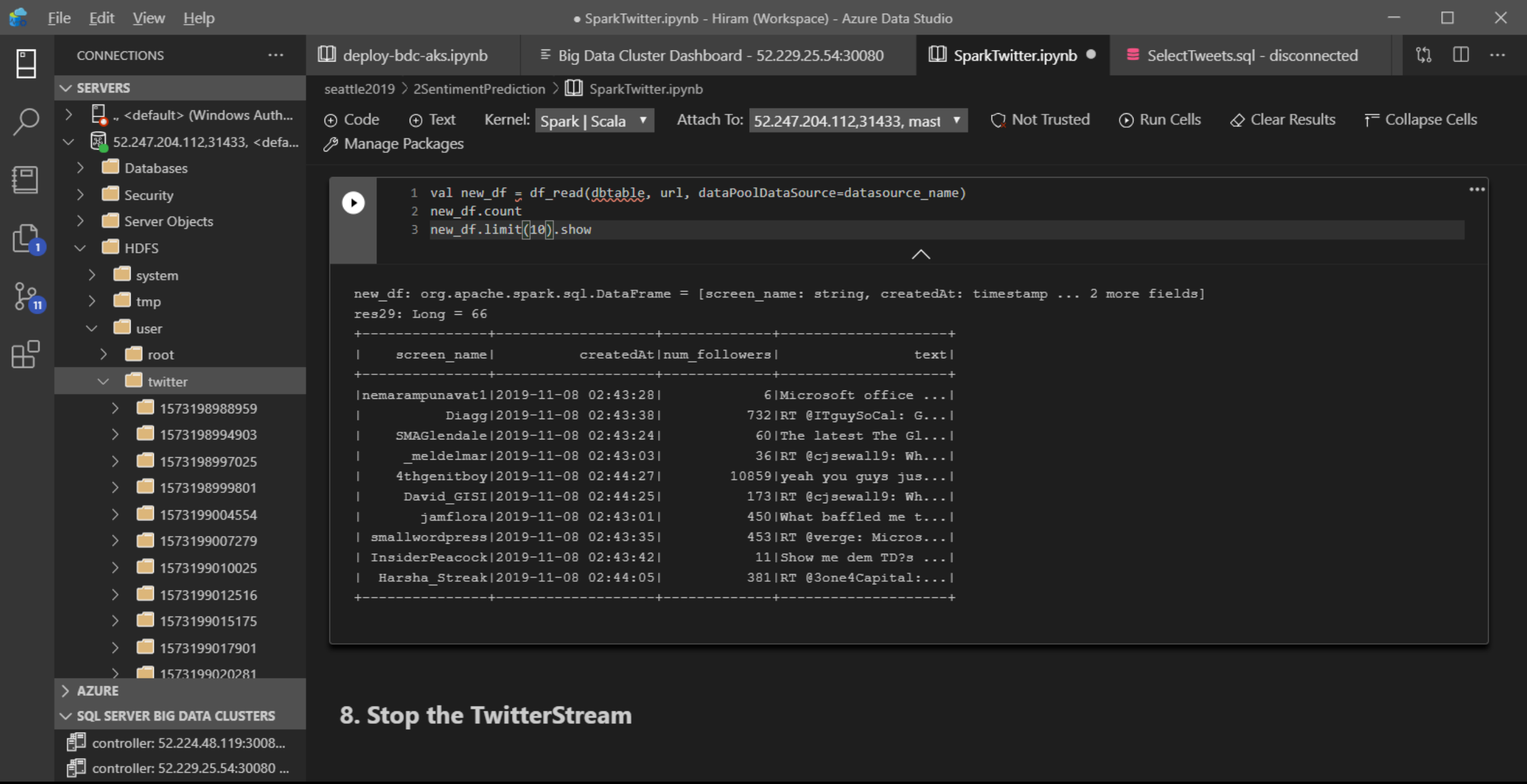
Thanks for writing in. The API does not return full-text reviews at this point; currently, the API will return 3 review snippets (the reviews that are currently displayed at the top when sorted by Yelp Sort, which is determined by recency, user voting, and other review quality factors to help consumers make informed decisions) of 160 characters. The Yelp API cannot be configured to return alternative review excerpts.

As a reminder, all API integrations must be consumer-facing (i.e. not for B2B dashboards or analytics) and must abide by our Terms of Use (https://www.yelp.com/developers/api_terms) and Display Requirements (https://www.yelp.com/developers/display_requirements). Lastly, all API responses cannot downloaded, stored, cached, or prefetched for more than 24 hours (except for the business ID for backend matching purposes) and scraping Yelp via any means is strictly prohibited.

Please let me know if you have other questions.

Thanks!

Learn more:
yelp.com/developers/api_terms
yelp.com/developers/display_requirements


File Edit View Help
• SparkTwitter.ipynb - Hiram (Workspace) - Azure Data Studio
CONNECTIONS
SERVERS
<default> (Windows Auth...
52.247.204.112,31433, <defa...
Databases
Security
Server Objects
HDFS
system
tmp
user
root
twitter
1573198988959
1573198994903
1573198997025
1573198999801
1573199004554
1573199007279
1573199010025
1573199012516
1573199015175
1573199017901
AZURE
SQL SERVER BIG DATA CLUSTERS
controller: 52.224.48.119:3008...
controller: 52.229.25.54:30080 ...
deploy-bdc-aks.ipynb
Big Data Cluster Dashboard - 52.229.25.54:30080
SparkTwitter.ipynb
SelectTweets.sql - disconnected
seattle2019 > 2SentimentPrediction > SparkTwitter.ipynb
Code
Text
Kernel: Spark | Scala
Attach To: 52.247.204.112,31433, mast
Not Trusted
Run Cells
Clear Results
Collapse Cells
Manage Packages
1 val new_df = df_read(dbtable, url, dataPoolDataSource=datasource_name)
2 new_df.count
3 new_df.limit(10).show
new_df: org.apache.spark.sql.DataFrame = [screen_name: string, createdAt: timestamp ... 2 more fields]
res29: Long = 66
+---------------+-------------------+------------+--------------------+
| screen_name| createdAt|num_followers| text|
+---------------+-------------------+------------+--------------------+
nemarampunavat1	2019-11-08 02:43:28	6	Microsoft office ...
Diagg	2019-11-08 02:43:38	732	RT @ITguySoCal: G...
SMAGlendale	2019-11-08 02:43:24	60	The latest The Gl...
_meldelmar	2019-11-08 02:43:03	36	RT @cjsewall9: Wh...
4thgenitboy	2019-11-08 02:44:27	10859	yeah you guys jus...
David_GISI	2019-11-08 02:44:25	173	RT @cjsewall9: Wh...
jamflora	2019-11-08 02:43:01	450	What baffled me t...
smallwordpress	2019-11-08 02:43:35	453	RT @verge: Micros...
InsiderPeacock	2019-11-08 02:43:42	11	Show me dem TD?s ...
Harsha_Streak	2019-11-08 02:44:05	381	RT @3one4Capital:...
+---------------+-------------------+------------+--------------------+
8. Stop the TwitterStream
Learn more:
SQL Server Samples - Big Data Clusters - Spark

File Edit View Help SelectTweets.sql - 52.247.204.112,31433.TwitterData (admin) - Hiram (Workspace) - Azure Data Studio

CONNECTIONS

SERVERS

., <default> (Windows Auth...

52.247.204.112,31433, <defa...

Databases

Security

Server Objects

HDFS

FleitasArts

AZURE

SQL SERVER BIG DATA CLUSTERS

controller: 52.224.48.119:3008...

controller: 52.229.25.54:30080 ...

SparkTwitter.ipynb

SelectTweets.sql - 52.247...(admin)

SandDance: SelectTweets.sql

52.247.204.11

seattle2019 > 2SentimentPrediction > SelectTweets.sql

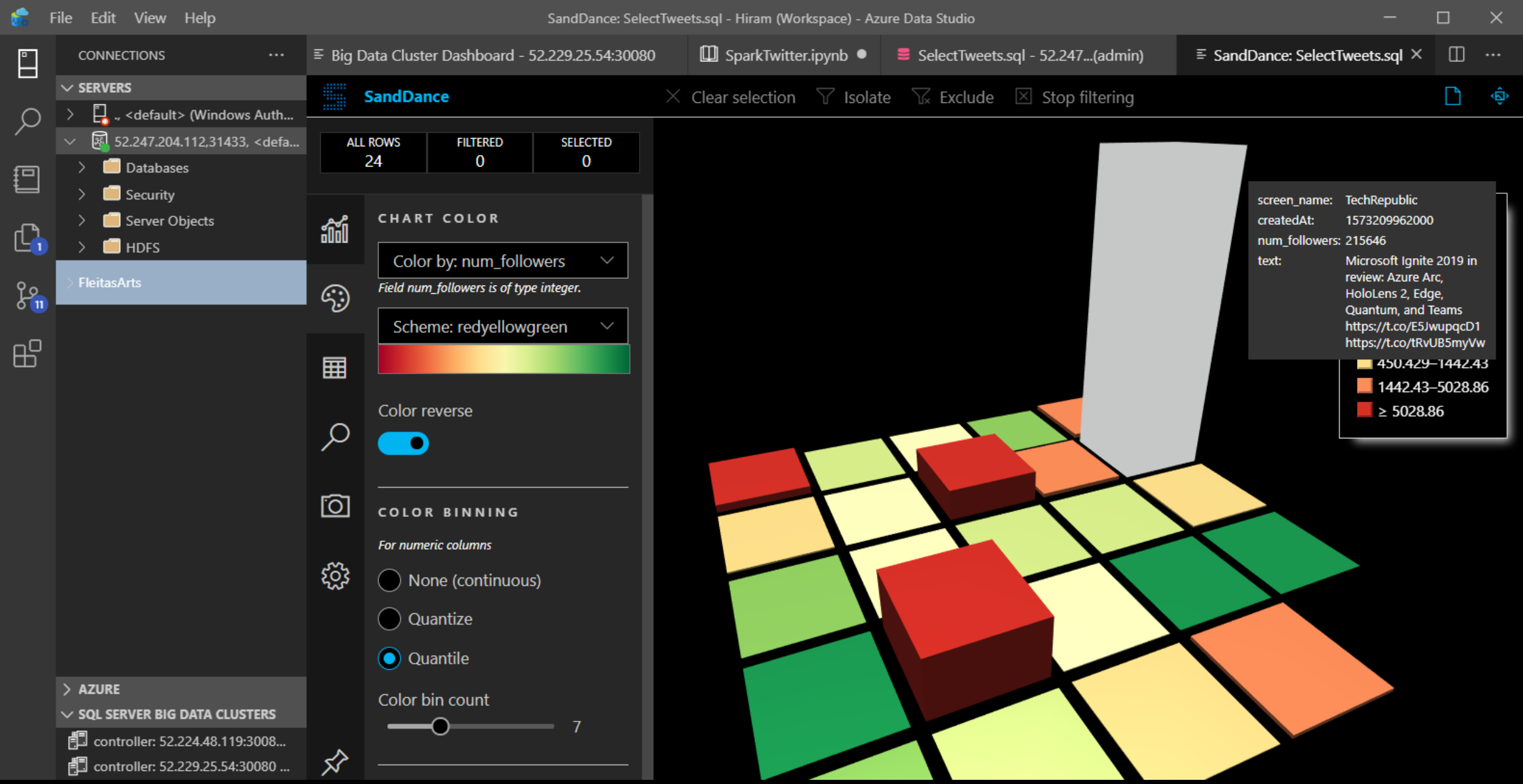
Run Cancel Disconnect Change Connection TwitterData Explain Enable SQLCMD

```
1  use twitterdata
2  go
3  select  top 100 *
4  from    dbo.tweets
5  where   screen_name = 'TechRepublic'
6  or      text like '%Microsoft%'
7  order by num_followers desc
8
```

Results Messages

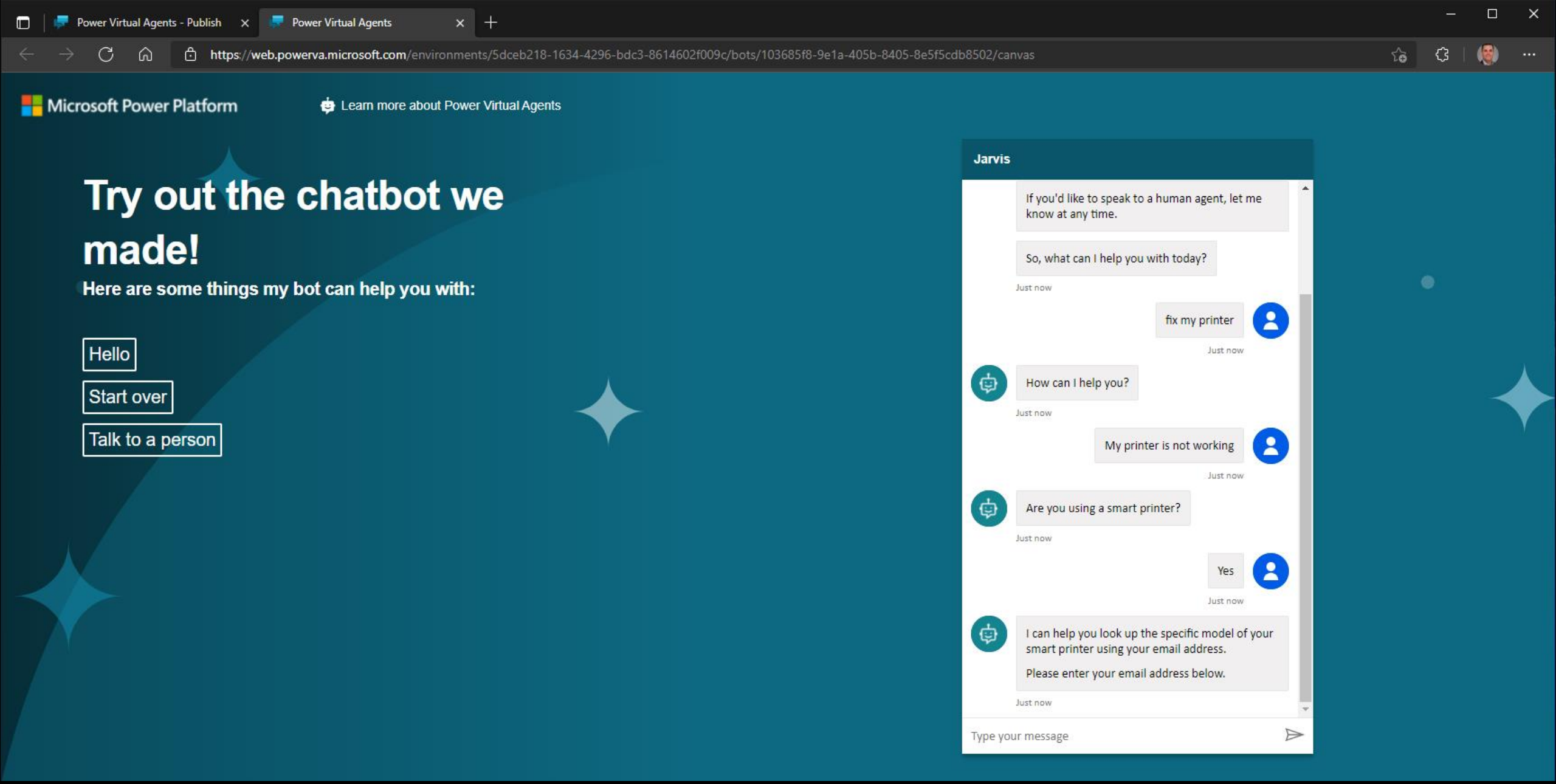
	screen_name	createdAt	num_followers	text
1	TechRepublic	2019-11-08 02:46:02.000	215646	Microsoft Ignite 2019 in rev…
2	hannytalker	2019-11-08 02:45:25.000	37628	The final edition for 2019 o…
3	PharmZay	2019-11-08 02:43:24.000	22096	RT @V_eektorrrr: I'm a fast …
4	AJBCSurveys	2019-11-08 02:44:08.000	5676	Look what's up: Microsoft So…
5	JRoosen	2019-11-08 02:44:41.000	3411	RT @JayTHL: any Microsoft pe…
6	Secnewsbytes	2019-11-08 02:47:17.000	2744	PayPal Upsets Microsoft as P…
7	uncle_dallas	2019-11-08 02:46:08.000	2155	RT @gcaughey: What does a Wi…
8	catrinphysio	2019-11-08 02:44:48.000	908	@DrDylanParry @amcunningham …
9	KiyaraSabel	2019-11-08 02:47:58.000	875	RT @FurlinNick: Who do you t…

Learn more:
SQL Server Samples - Big Data Clusters - Spark



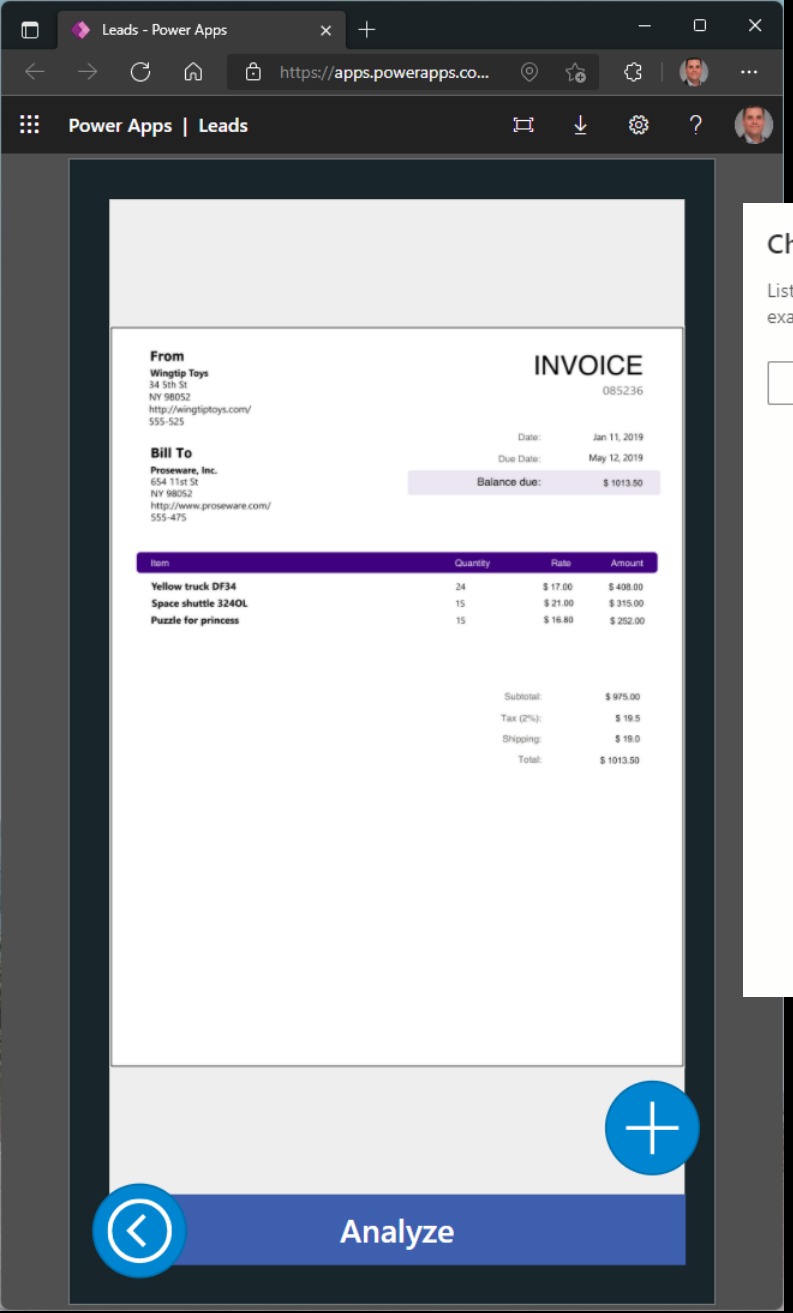
Learn more:
[SQL Server Samples - Big Data Clusters - Spark]

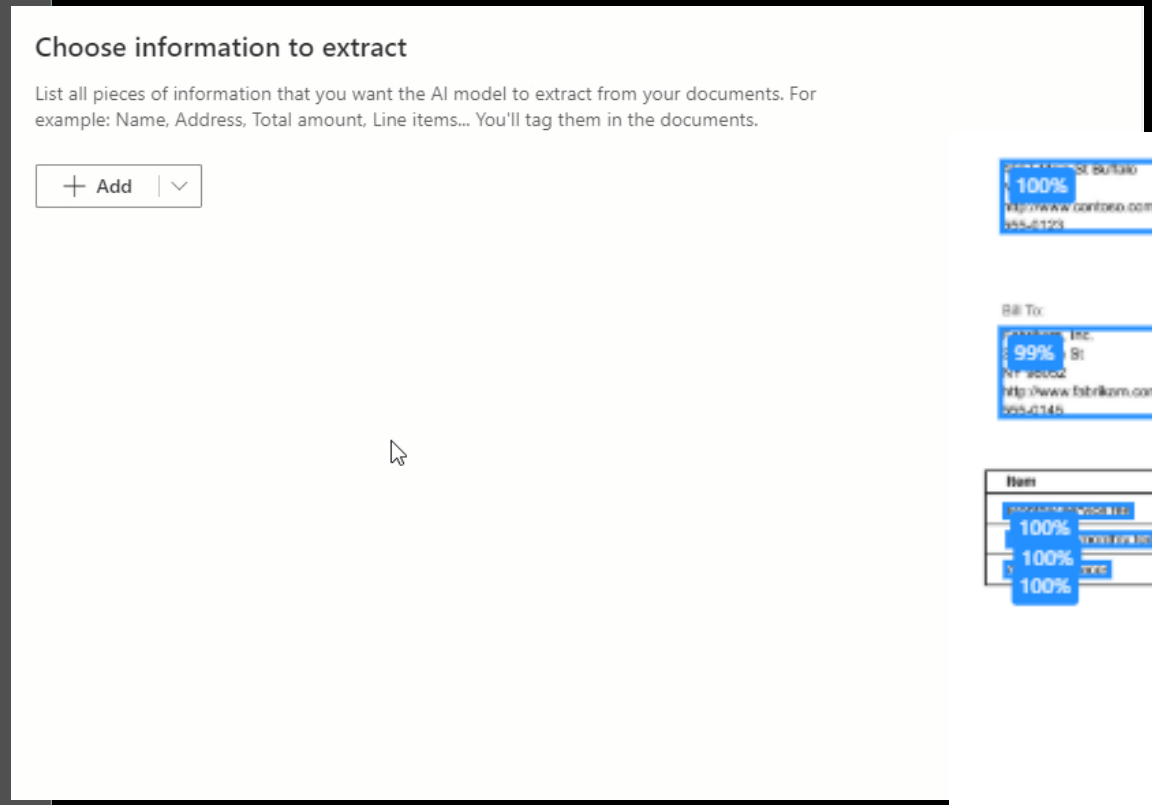
QnA Chat Bots $200K

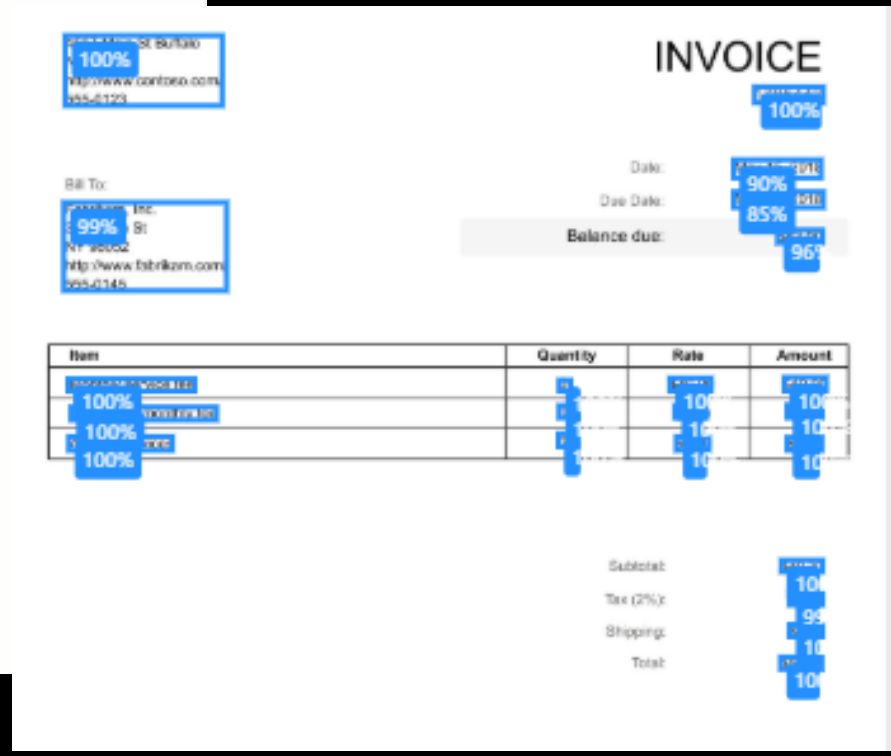


Learn more:
[PVA - Build Printer Troubleshooting Bot]

Forms $2M







Learn more:
[Form Processing - Create first model]

Home
Learn
Apps
Create
Data
Business logic
AI Builder
Build
Models
Solutions

Refine a model for your business needs

Category classification

Categorize text by its meaning so it's easier to analyze.

Form processing

Read and save information from standard documents.

Object detection

Recognize and count things in images.

Prediction

Predict whether something will happen.

Entity extraction

Recognize specific information about your business from data.

Get straight to productivity

Business card reader

Automatically process business card information

Category classification

Categorize text by its meaning so it's easier to analyze.

Entity extraction

Extract basic information from your data

Key phrase extraction

Extract the key talking points from text

Language detection

Receipt processing

Sentiment analysis

Text recognition

Capital Modeling $500M

Digital Transformation
We are people, not interfaces

Before

- 7 User touchpoints
- 4 Unique manual processes
- Many different files and folders
- Significant user downtime

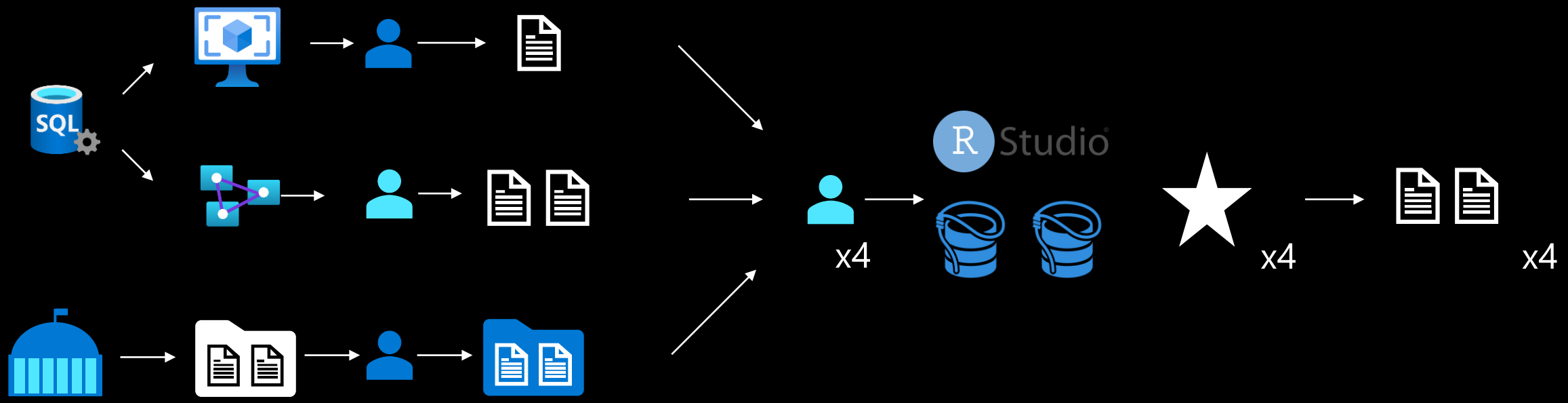
After

- 1 Single pipeline
- 3 Primary stored procedures
- 1 SQL Server
- Minimal user downtime

Digital Transformation

We are people, not interfaces

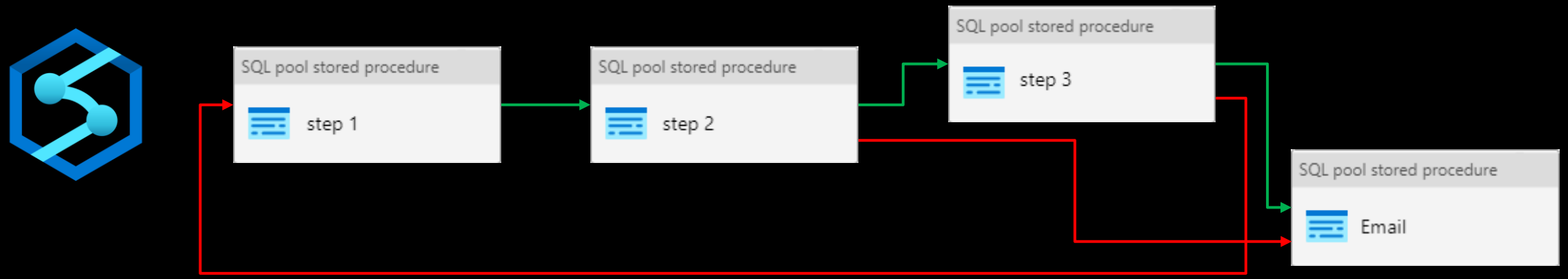
Before



Digital Transformation

We are people, not interfaces

After

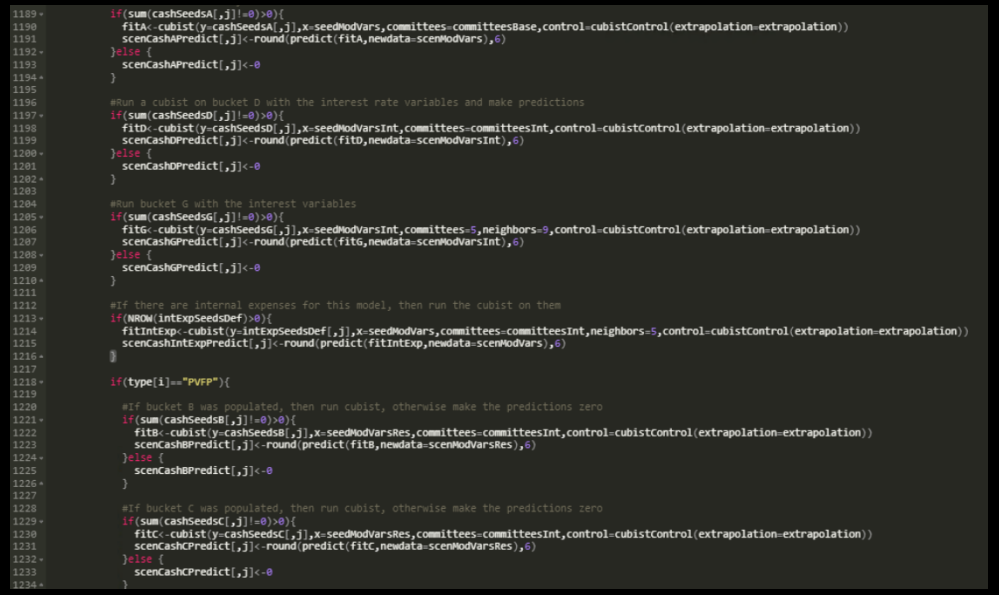


Digital Transformation

We are people, not interfaces

Before

- Loop based logic
- WET (Write Everything Twice)
- Inefficient
- Frequent R troubleshooting



```
        if(sum(cashSeedsA[,j]!=0)>0){
          fitA<-cubist(y=cashSeedsA[,j],x=seedModVars,committees=committeesBase,control=cubistControl(extrapolation=extrapolation))
          scenCashAPredict[,j]<-round(predict(fitA,newdata=scenModVars),6)
        }else {
          scenCashAPredict[,j]<-0
        }

        #Run a cubist on bucket D with the interest rate variables and make predictions
        if(sum(cashSeedsD[,j]!=0)>0){
          fitD<-cubist(y=cashSeedsD[,j],x=seedModVarsInt,committees=committeesInt,control=cubistControl(extrapolation=extrapolation))
          scenCashDPredict[,j]<-round(predict(fitD,newdata=scenModVarsInt),6)
        }else {
          scenCashDPredict[,j]<-0
        }

        #Run bucket G with the interest variables
        if(sum(cashSeedsG[,j]!=0)>0){
          fitG<-cubist(y=cashSeedsG[,j],x=seedModVarsInt,committees=5,neighbors=9,control=cubistControl(extrapolation=extrapolation))
          scenCashGPredict[,j]<-round(predict(fitG,newdata=scenModVarsInt),6)
        }else {
          scenCashGPredict[,j]<-0
        }

        #If there are internal expenses for this model, then run the cubist on them
        if(NROW(intExpSeedsDef)>0){
          fitIntExp<-cubist(y=intExpSeedsDef[,j],x=seedModVars,committees=committeesInt,neighbors=5,control=cubistControl(extrapolation=extrapolation))
          scenCashIntExpPredict[,j]<-round(predict(fitIntExp,newdata=scenModVars),6)
        }

        if(type[i]=="PVFP"){

          #If bucket B was populated, then run cubist, otherwise make the predictions zero
          if(sum(cashSeedsB[,j]!=0)>0){
            fitB<-cubist(y=cashSeedsB[,j],x=seedModVarsRes,committees=committeesInt,control=cubistControl(extrapolation=extrapolation))
            scenCashBPredict[,j]<-round(predict(fitB,newdata=scenModVarsRes),6)
          }else {
            scenCashBPredict[,j]<-0
          }

          #If bucket C was populated, then run cubist, otherwise make the predictions zero
          if(sum(cashSeedsC[,j]!=0)>0){
            fitC<-cubist(y=cashSeedsC[,j],x=seedModVarsRes,committees=committeesInt,control=cubistControl(extrapolation=extrapolation))
            scenCashCPredict[,j]<-round(predict(fitC,newdata=scenModVarsRes),6)
          }else {
            scenCashCPredict[,j]<-0
          }
```

After

- Functional programming
- DRY (Don’t Repeat Yourself)
- Little need for R troubleshooting

```
trainCubist     <- function(dt, com = 15, extp = 90){Cubist::cubist(y = getY(dt),x = getX(dt),committees = getCom(dt),control =
predictCubist   <- function(aModel, dt){stats::predict(aModel, newdata = getX(dt))}
```

SQLEngine_AzureDataStudio_Sample.ipynb •

ImpactSOA_repo > SOA-BackToBasicsAnalyticsForActuaries > ImpactSOA_CodeSamples > SQLEngine_AzureDataStudio_Sample.ipynb

+ Cell | Run all | Kernel SQL | Attach to

[21]
```
1  CREATE EXTERNAL LIBRARY Cubist  FROM (CONTENT = 'E:/ServerPackageDirectory/Cubist_0.2.3.zip'  ) WITH (LANGUAGE = 'R');
2  CREATE EXTERNAL LIBRARY mlbench FROM (CONTENT = 'E:/ServerPackageDirectory/mlbench_2.1-1.zip' ) WITH (LANGUAGE = 'R');
```

Commands completed successfully.

Total execution time: 00:00:00.906

[23]
```
1  exec sp_execute_external_script
2      @language = N'R',
3      @script = N'
4  library(Cubist)
5  library(mlbench)
6  data(BostonHousing)
7
8  ## 1 committee, so just an M5 fit:
9  mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
10 print(mod1)
11
12 ## Now with 10 committees
13 mod2 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv, committees = 10)
14 print(mod2)
15 ';
```

STDERR message(s) from external script: Warning messages: 1: package 'Cubist' was built under R version 3.5.3 2: package 'mlbench' was built under R version 3.5.3

STDOUT message(s) from external script: Call: cubist.default(x = BostonHousing[, -14], y = BostonHousing$medv) Number of samples: 506 Number of predictors: 13 Number of committees: 1 Number of rules: 4 Call: cubist.default(x = BostonHousing[, -14], y = BostonHousing$medv, committees = 10) Number of samples: 506 Number of predictors: 13 Number of committees: 10 Number of rules per committee: 4, 6, 4, 6, 6, 7, 7, 7, 4, 5

Total execution time: 00:00:02.440

How to get started? How to reproduce?

1. Identify manual data processes & transformations of high value (ROI).
2. Simplify by consolidating workloads (leverage cloud).
3. Democratize understanding by sharing knowledge.
4. Accelerate business value for down-stream consumers/users.
5. aka.ms/mlsqldev
6. aka.ms/adf/azurelearn
7. RStudio Cheatsheets - RStudio

Length of Stay $40M

Realtime Scoring

Proc (errors), Train, Score

In [3]:

```
-- My simple proc to serialize the model bin, cause train_model_real_time_scoring errors.
Use Hospital_Py
go
create or alter proc [GetRTSModelRF]
as
declare @info varbinary(max);
select @info = info from dbo.ColInfo;
declare @info varbinary(max);
select @info = info from dbo.ColInfo;

exec sp_execute_external_script @language = N'Python', @script = N'
import dill
from numpy import sqrt
from pandas import DataFrame
from revoscalepy import rx_set_compute_context, RxSqlServerData, rx_dforest, RxOdbcData, rx_serialize_model, rx_write_object, RxLocalSeq
from microsoftml import adadelta_optimizer

connection_string = "Driver=SQL Server;Server=localhost;Database=Hospital_Py;Trusted_Connection=true;"

column_info = dill.loads(info)

##      Set training dataset, set features and types.

variables_all = [var for var in column_info]
#variables_to_remove = ["eid", "vdate", "discharged", "facid"]
variables_to_remove = ["ClaimClaimID", "ClaimDateClosed", "ClaimReportedDate"]
training_variables = [x for x in variables_all if x not in variables_to_remove]
LoS_Train = RxSqlServerData(sql_query = "SELECT ClaimClaimID, {} FROM LoS WHERE ClaimClaimID IN (SELECT ClaimClaimID from Train_Id)".form
at(", ".join(training_variables)),
                            connection_string = connection_string,
                            column_info = column_info)

##      Specify the variables to keep for the training

#variables_to_remove = ["eid", "vdate", "discharged", "facid", "lengthofstay"]
variables_to_remove = ["ClaimClaimID", "ClaimDateClosed", "ClaimReportedDate", "lengthofstay"]
training_variables = [x for x in variables_all if x not in variables_to_remove]
formula = "lengthofstay ~ " + " + ".join(training_variables)

## Train RF Model
dest = RxOdbcData(connection_string, table = "RTS")
model = rx_dforest(formula=formula,
                   data=LoS_Train,
                   n_tree=40,
                   cp=0.00005,
                   min_split=int(sqrt(70000)),
                   max_num_bins=int(sqrt(70000)),
                   seed=5)
serialized_model = rx_serialize_model(model, realtime_scoring_only = True)
rx_write_object(dest, key_name="id", key="RF", value_name="value", value=serialized_model, serialize=False, compress=None, overwrite=Fals
e)'

, @params = N'@info varbinary(max)'
, @info = @info;

GO
```

Learn more:
LoS - Realtime Scoring

```
exec GetRTSModelRF;

-- Dev server: Total execution time: 00:20:17.325
```

In [8]:

```
select id, (datalength(value)/1024)/1024 as MB from RTS
```

(1 row affected)

Total execution time: 00:00:00.010

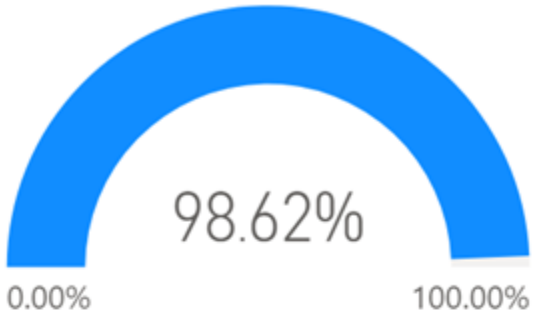
Out[8]:

id	MB
RF	0

```
--- Real Time Scoring

--      Get the trained model
DECLARE @model_name VARCHAR(3) = 'RF'
DECLARE @model VARBINARY(max) = (SELECT value FROM [dbo].[RTS] WHERE id = @model_name);

--- Real Time Scoring is meant for small scoring request, which is why we select the top 10 for this example.
DECLARE @inputData VARCHAR(max);
SET @inputData = 'SELECT TOP (10) ClaimClaimID, ClaimClaimStatusID, ClaimStatusDescription, StatesStateCode, LossTypeDescription, PolicyVersi
        PolicyVersionAttributesOccupancyType, PolicyVersionAttributesCoverageA, ClaimMoneyLosses, ClaimMoneyLAE, ClaimRoomsWithDamage,
        number_of_issues, lengthofstay
FROM LoS WHERE  ClaimClaimID NOT IN (SELECT ClaimClaimID from Train_Id) ORDER BY ClaimClaimID';

DECLARE @output_table TABLE(lengthofstay_Pred FLOAT);
INSERT @output_table EXEC [dbo].[sp_rxPredict] @model = @model, @inputData = @inputData;
DROP TABLE IF EXISTS RTS_PredictionBak;
exec sp_rename 'RTS_Prediction', 'RTS_PredictionBak';
SELECT * INTO RTS_Prediction FROM @output_table
```

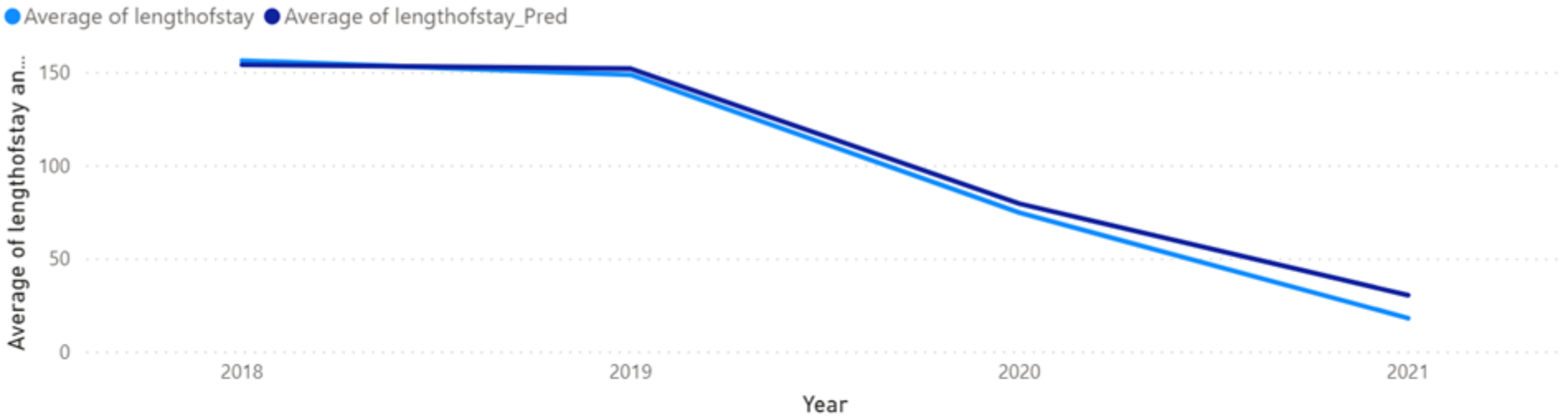
Learn more:
[LoS - Realtime Scoring]

Confidence

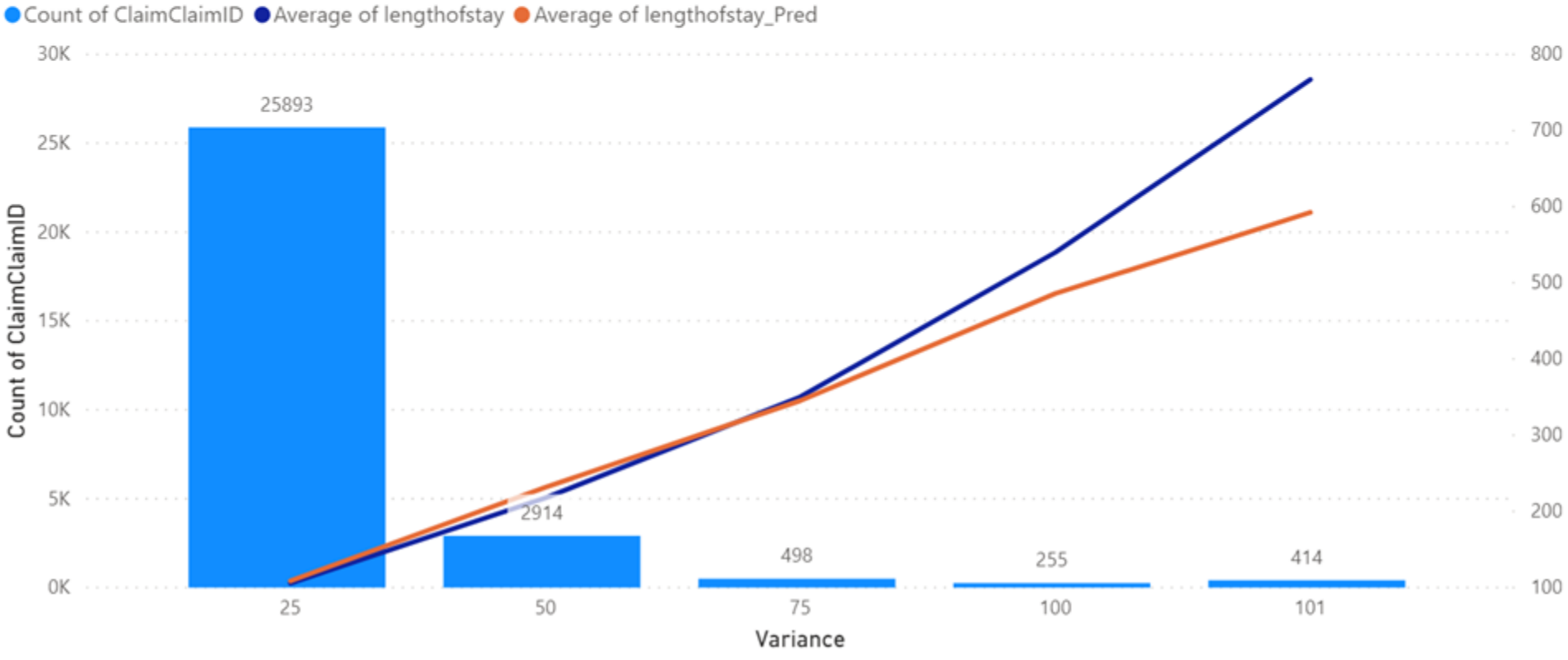


ClaimClaimID	lengthofstay	lengthofstay_Pred
[redacted]	4.00	30.56
[redacted]	4.00	20.58
[redacted]	5.00	20.66
[redacted]	3.00	15.41
[redacted]	3.00	15.41
[redacted]	6.00	15.71
[redacted]	6.00	15.41
[redacted]	16.00	28.67
[redacted]	14.00	18.32
[redacted]	15.00	26.80
[redacted]	10.00	15.83
[redacted]	5.00	16.71
[redacted]	13.00	26.99
[redacted]	15.00	27.40
[redacted]	16.00	20.40
[redacted]	16.00	51.16
[redacted]	11.00	18.18
[redacted]	9.00	42.67
[redacted]	13.00	48.96
[redacted]	5.00	16.73
[redacted]	16.00	19.11
[redacted]	16.00	38.60
[redacted]	17.00	40.22

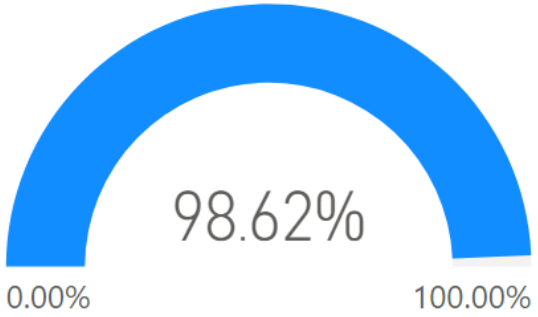
Average of lengthofstay and Average of lengthofstay_Pred by Year



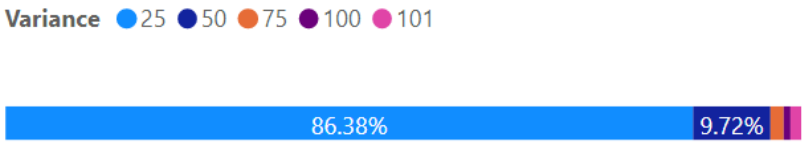
Count of ClaimClaimID, Average of lengthofstay and Average of lengthofstay_Pred by Variance



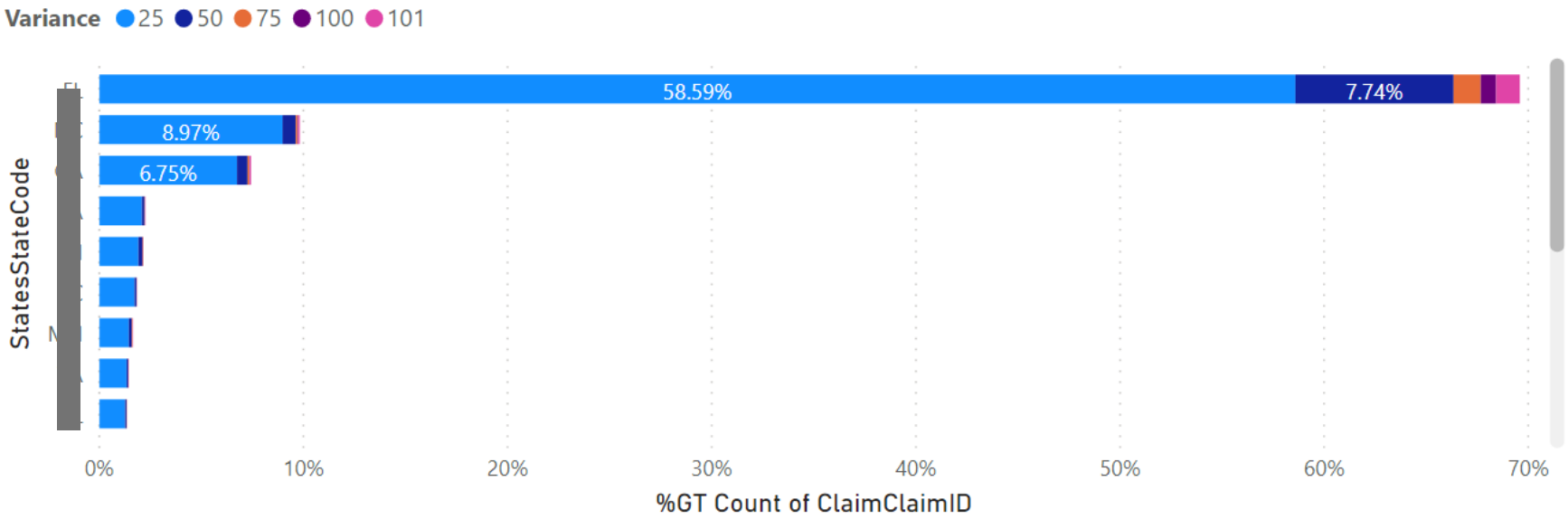
Confidence



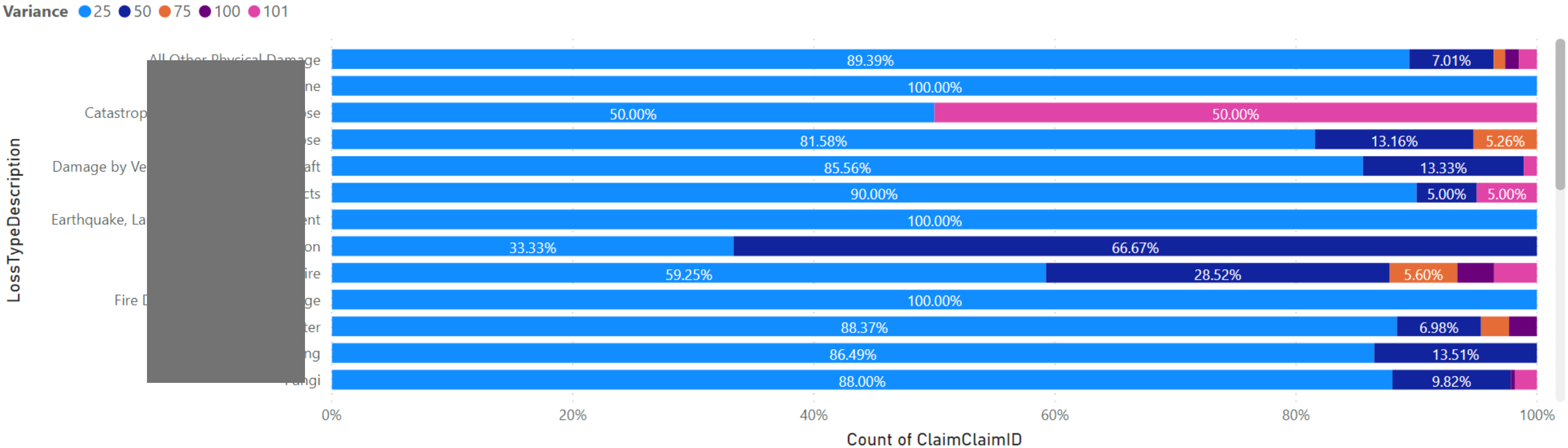
%GT Count of ClaimClaimID by Variance



%GT Count of ClaimClaimID by StatesStateCode and Variance



Count of ClaimClaimID by LossTypeDescription and Variance



Search by Photo $200K

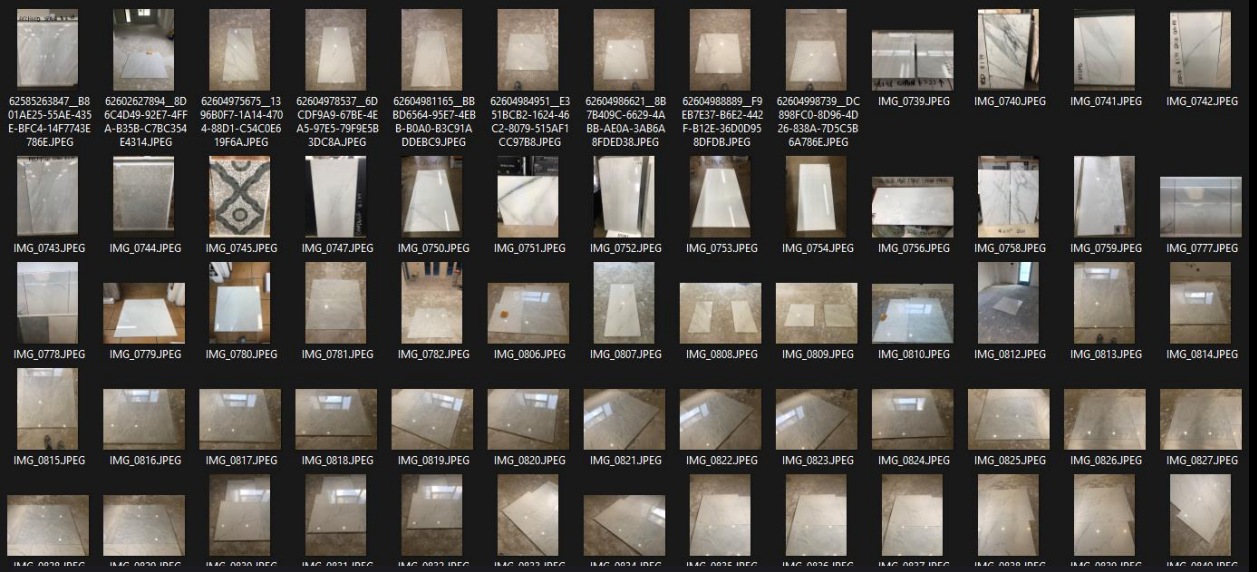
Power Apps

SRM - Power Apps

https://apps.powerapps.com/play/02a246dc-4251-46c4-95a9-200c4f210af0?tenantId=800d472c-8288-4f27-8978-f726a7a3d1f0&source=portal&screenColor=rgba(76%2C%2074%2C%2072%2C%201)

Power Apps | SRM

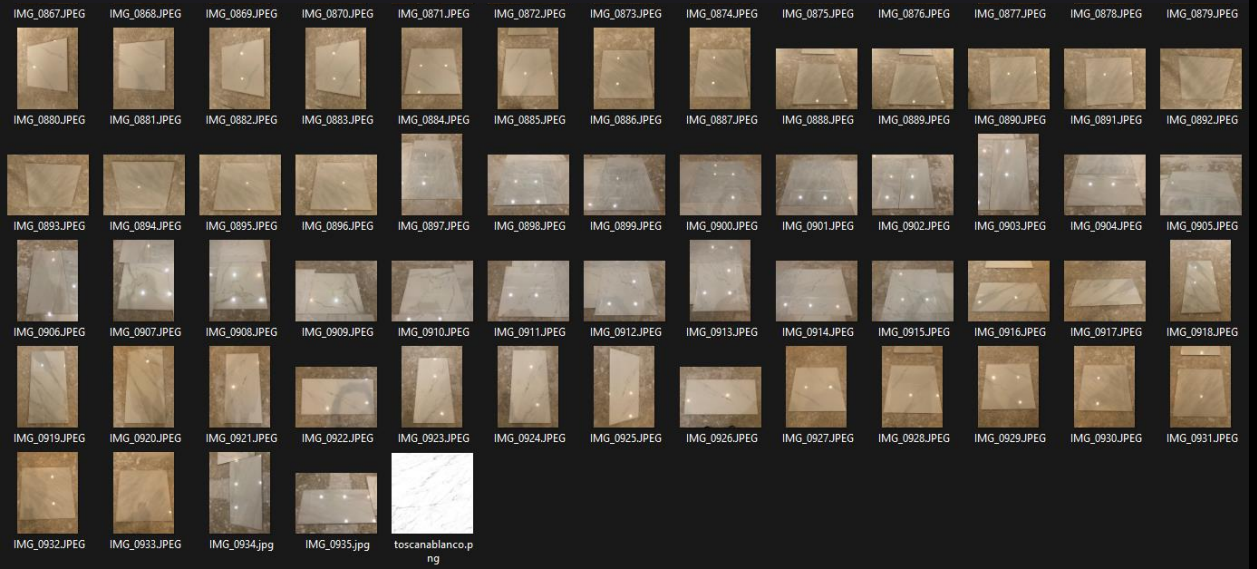
Search by Photo

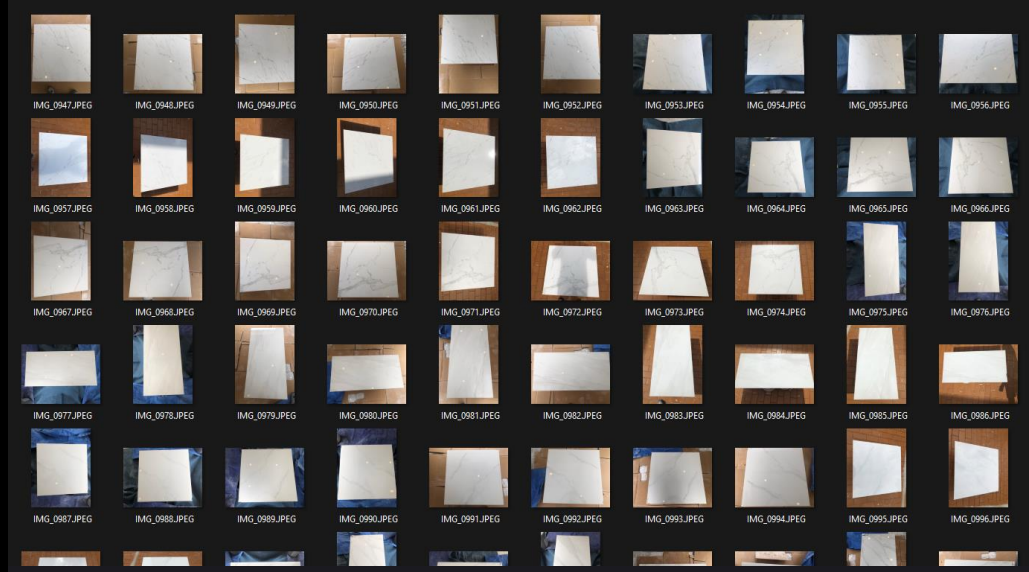


~1 background

161 photos, 15 angles

(Fail)

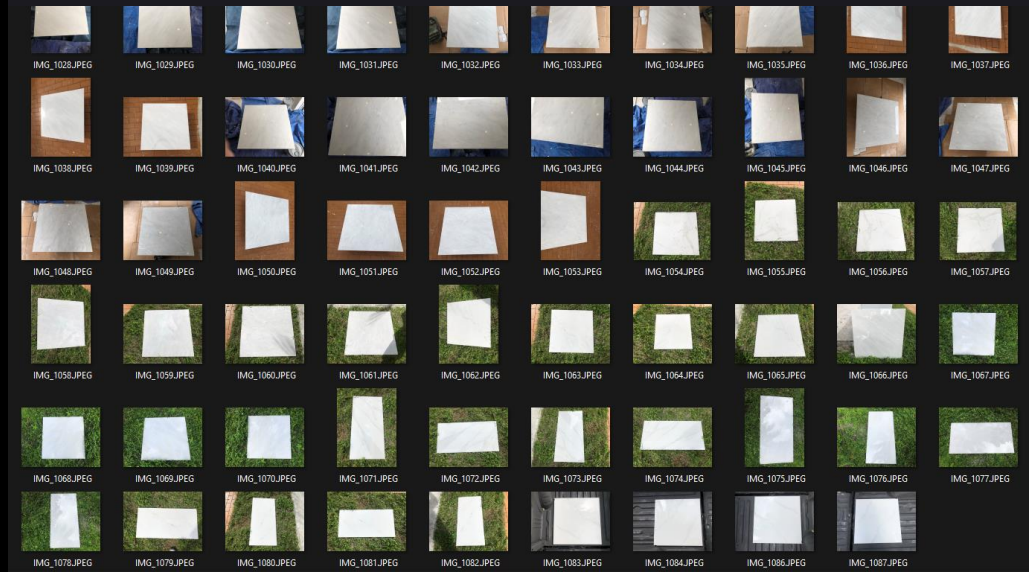


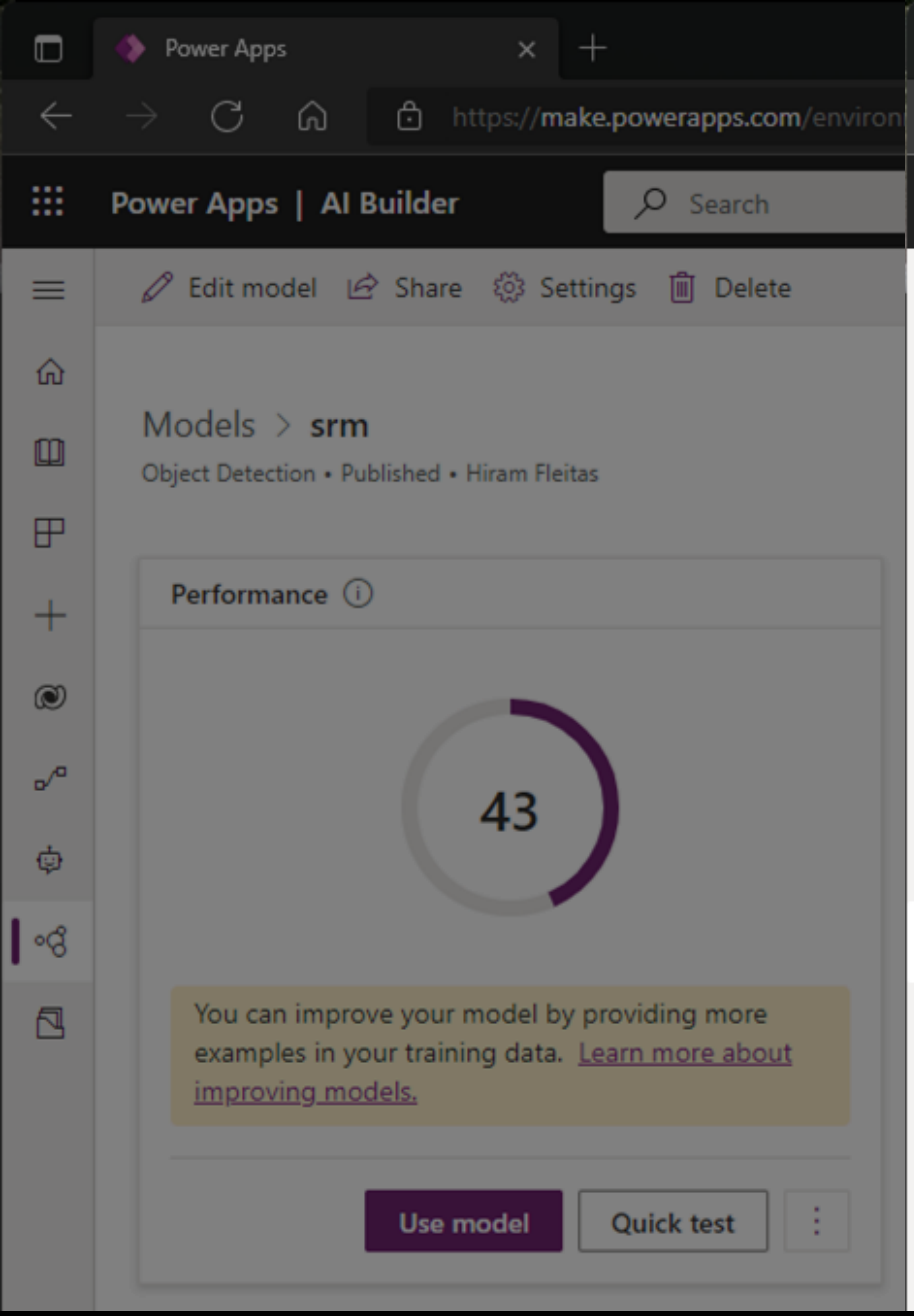


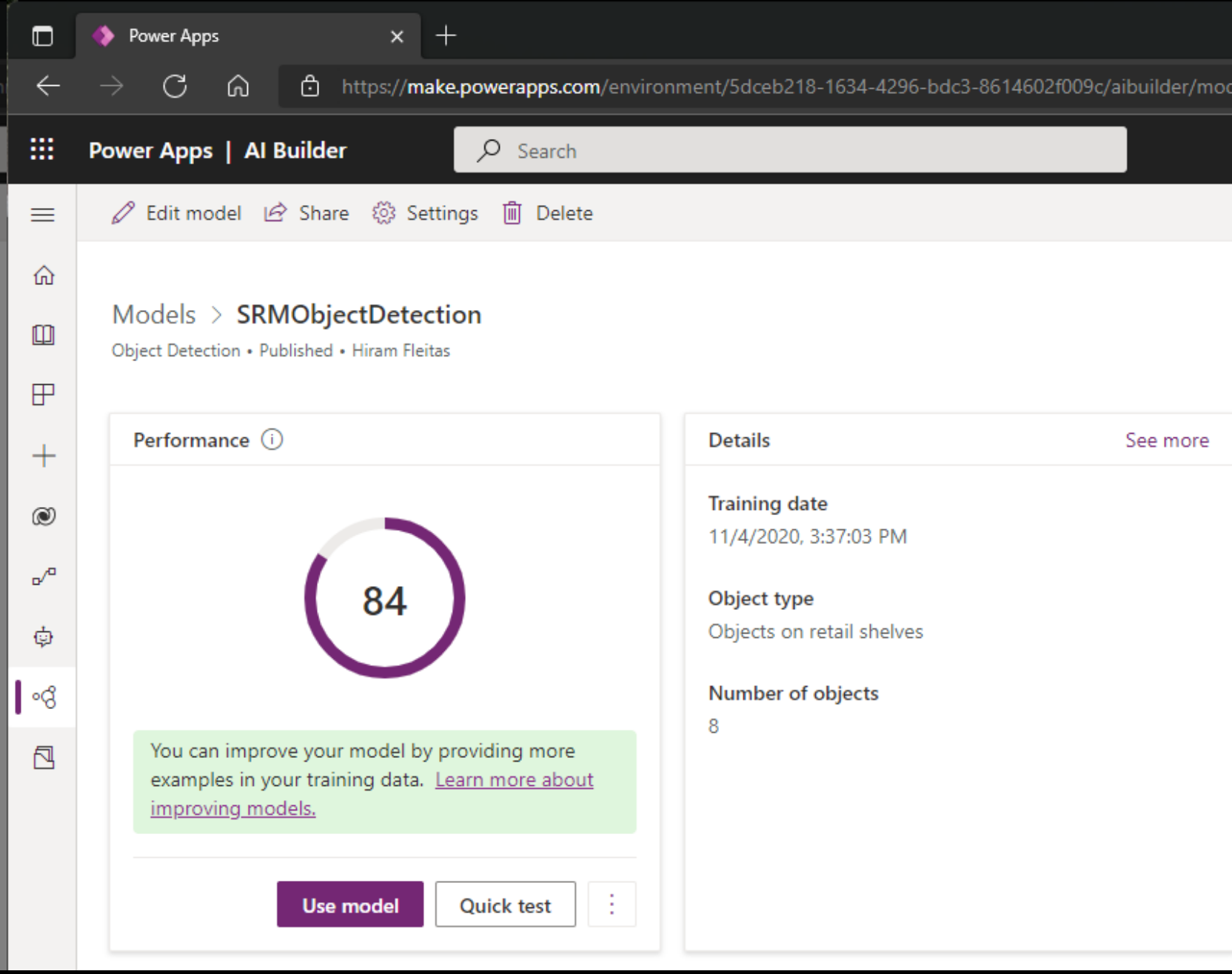
5 backgrounds

139 photos, 4 angles

(Successful)



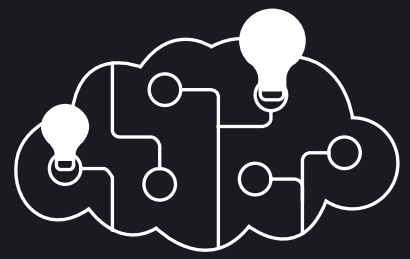




Learn more:
[Use more diverse images]
[Interpret performance score]

Summary

1. Sentiment
2. QnA Chat Bots
3. Forms
4. Capital Modeling
5. Length of Stay
6. Search by Photo



1 2 3 4 5 6 7 8

Iris Ski Text Forms Image LoS Price ∞

Additional Resources

1. Repo1 (SQL, Python, JS on Node, Power BI)
2. Repo2 (SQL BDC, Spark/Scala, Python, Azure Cog API, Paginated Reports)
3. Repo3 (Synapse, DevOps, In-DB ML, R, Purview)
4. Video (3 minutes)
5. Video (1 hour)

Microsoft

Thank you

Hiram Fleitas

aka.ms/hiram