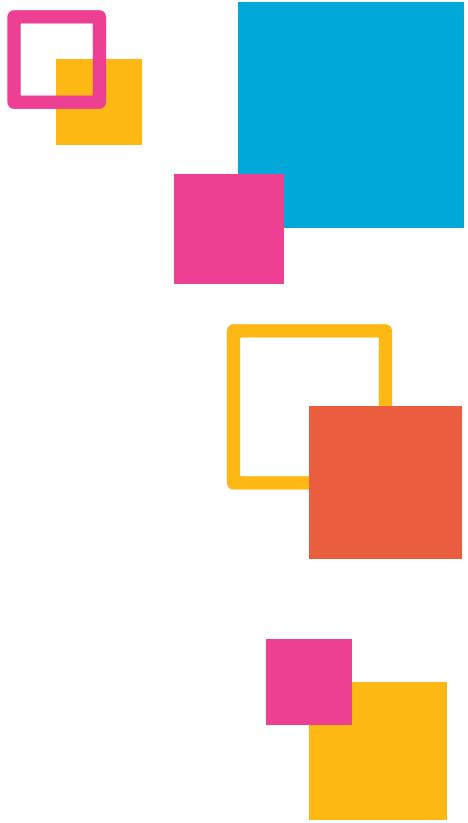
#JOINdata

@lookerdata



Hands-on Lab: Data Science in Action with Looker & R

looker

Shingi Samudzi
Professional Services

#JOINdata
@lookerdata

looker.com/hol

Select the **Data Science in Action with Looker & R** lab in the drop-down

Shingi Samudzi

Consultant, Professional Services

looker

Agenda

Introduction

Looker & R

Use cases

Exercise: Looker & R in action

Questions

The journey to data science with Looker

Is this right?

All data is stored on various disconnected Excel spreadsheets or databases

Are we tracking that?

Building clean data pipelines for Looker to model all data and be the single source of truth gives visibility to what is actually being tracked/measured and how

What does it really mean?

Simple statistical modeling helps create a picture of good vs bad conclusions to draw from data

What will happen tomorrow?

Strong modeling with minimized error allows for predictive analytics

looker

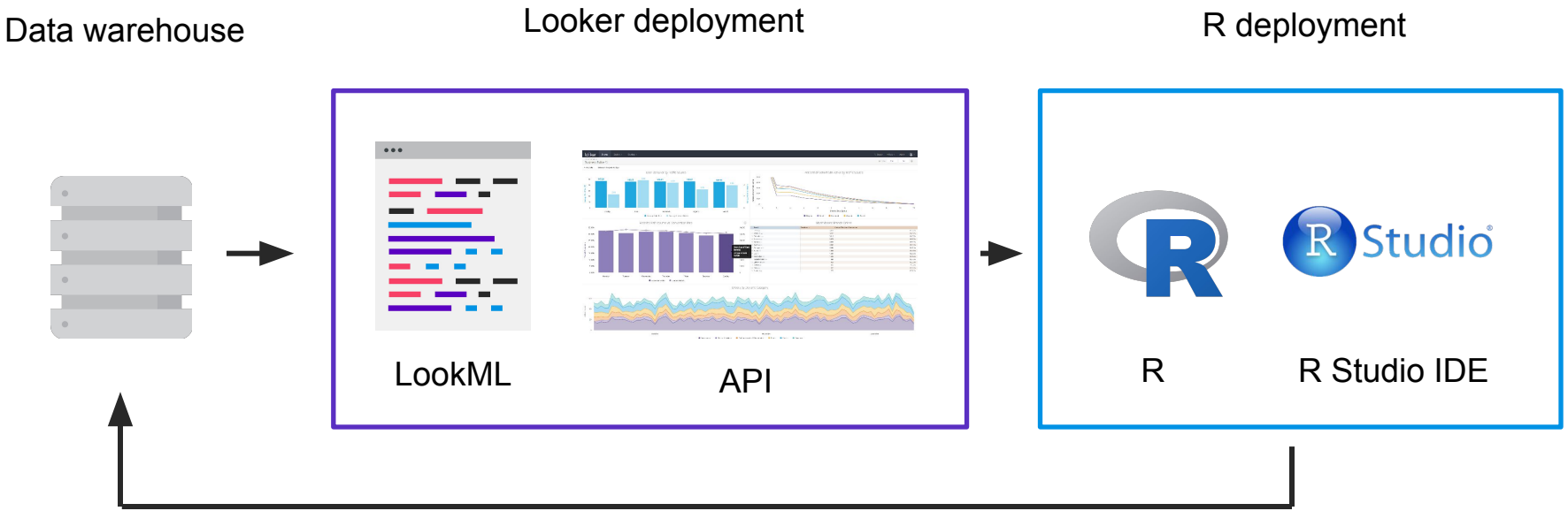
What is data science, really?

Software + Statistics + Massive Datasets

Using software to scale the application of statistical models and tests onto massive datasets

Can also include the data engineering that takes origin data thru ETL into a form appropriate for actual analysis

Looker + R system architecture



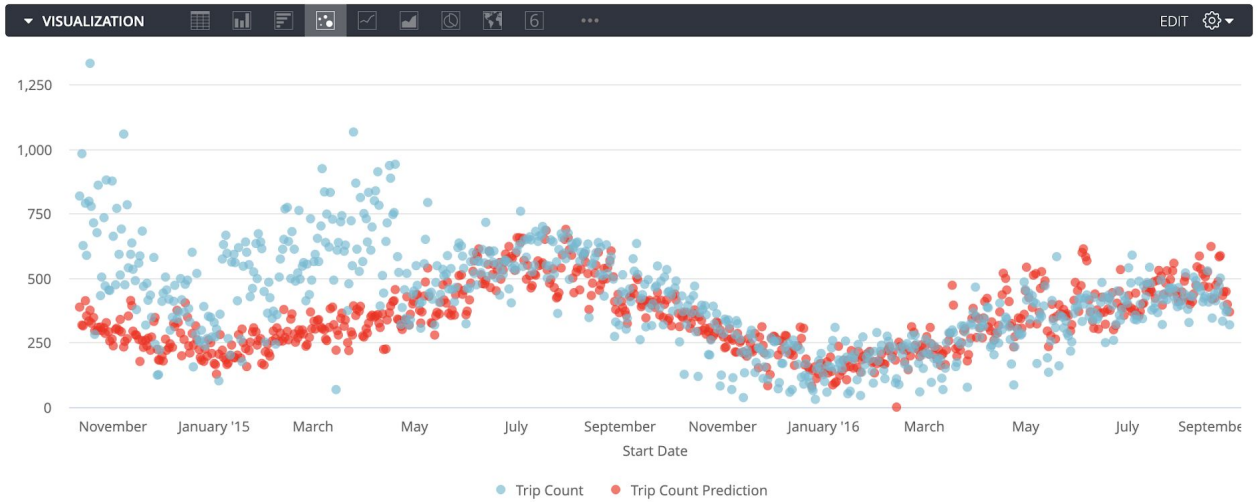
Use cases for Looker + R

- Predict a new customer's shopping choices based on demographic characteristics
- Predict how a customer will respond to a new product offering based on historical data
- Identify credit card fraud among millions of customer transactions
- Create inventory forecasts, financial projections, or event predictions from time series data

Let’s see Looker & R in action

Use cases for Looker + R

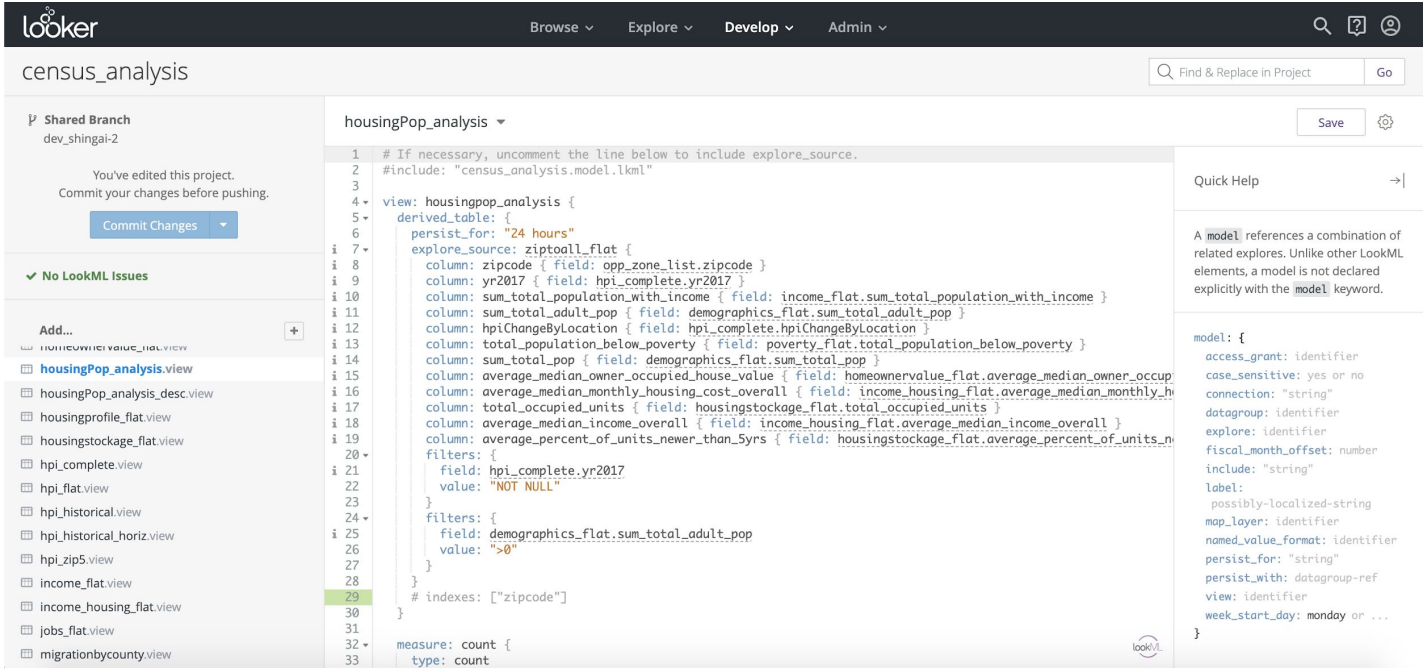
- Driving alerts and actions through Looker based on predicted outcomes
- Flex staff scheduling based on predicted customer volume in a store



looker

Step 1 — Model data in Looker

Use PDTs to build the model for Looks to ensure data pulled via API is the most up-to-date relative to your underlying dataset



looker

Step 2 — Using LookR to access Looks

Create a `looker.ini` file to store your API connection details

```
[Looker]
# API version is required
api_version=3.0
# Base URL for API. Do not include /api/* in the url
base_url=https://data.asoba.co:19999
# API 3 client id
client_id=v53XyrTCYKrqxkRDwt4y
# API 3 client secret
client_secret=J949yjq7hP5x8ggkSQWmW2Xx
# Optional embed secret for SSO embedding
#embed_secret=your_embed_SSO_secret
# Optional user_id to impersonate
user_id=1
# Set to false if testing locally against self-signed certs. Otherwise leave True
verify_ssl=True
```

Then install the LookR R package and establish a connection with your Looker instance

```
#Installing Lookr from dev branch
#devtools::install_github("looker/lookr", ref = "dev", force=TRUE)

#Initialize the LookerAPI connection
setwd("~/Projects/looker-sdk/")

library(lookr)

sdk <- LookerSDK$new(configFile = "looker.ini")

```


looker

Step 3 — Build/train ML model

Load libraries you plan on using, and then use the API to load data from your Look as a dataframe

```
#Make sure that the Looker Connection is initialized
source(file="~/Projects/initializeLooker.R")

setwd("~/Projects/DemandModeling")
#Load predictive R library
source(file="lib/predictiveR2.R")
source(file="lib/crossvalidation.R")

#Load all of the key libraries
library (plyr)
library(dplyr)
library(quantmod)
library(purrr)
library(data.table)
library(tidyr)
library(MVLM)
library(car)
library(ggplot2)
library(GGally)
library(scatterplot3d)
library(rrr)
library(caret)
library(leaps)

#pull data from the Looker API
base_data <- sdk$runLook(lookId = 51)

```

Step 3 — Build/train ML model

Load libraries you plan on using, and then use the API to load data from your Look as a dataframe

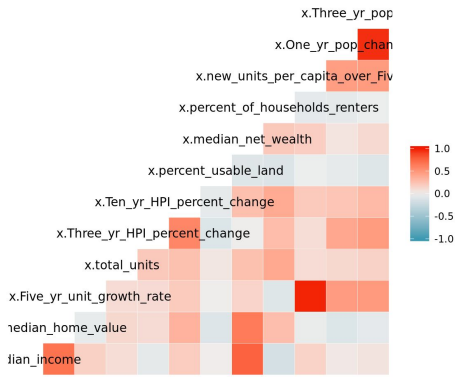
```
#Make sure that the Looker Connection is initialized
source(file="~/Projects/initializeLooker.R")

setwd("~/Projects/DemandModeling")
#Load predictive R library
source(file="lib/predictiveR2.R")
source(file="lib/crossvalidation.R")

#Load all of the key libraries
library (plyr)
library(dplyr)
library(quantmod)
library(purrr)
library(data.table)
library(tidyr)
library(MVLM)
library(car)
library(ggplot2)
library(GGally)
library(scatterplot3d)
library(rrr)
library(caret)
library(leaps)

#pull data from the Looker API
base_data <- sdk$runLook(lookId = 51)

```

Step 3 — Build/train ML model

Start with an exploratory data analysis to narrow down the set of variables you will use for your model

EDA techniques like `ggpairs` allow you to spot highly correlated variables

```
106  model_plot <- ggpairs(data=base_df2, columns=1:11, title="Real Estate Demand Variables",
     progress=NULL)
107
108  #second form of variable analysis to assess correlations between predictor variables
109
110  model_x <- base_df2 %>%
111      select(starts_with("x"))
112  model_y <- base_df2 %>%
113      select(starts_with("y"))
114
115  GGally::ggcorr(model_x)
116  GGally::ggcorr(model_y)
```



looker

Step 3 — Build/train ML model

Once you have narrowed down your variables, you can use a technique like Best Subsets Regression to measure regression model quality. In this example, I am measuring using three different metrics - Adjusted R^2, Bayesian information criteria, and Mallows Cp

```
237  #for 1yr HPI Change
238  var_subset2 <- regsubsets(y.One_yr_HPI_percent_change~., data = model_x4, nvmax = 14)
239  res_subset2<-summary(var_subset2)
240  res_sum2<-data.frame(
241    Adj.R2 = which.max(res_subset2$adjr2),
242    CP = which.min(res_subset2$cp),
243    BIC = which.min(res_subset2$bic)
```


looker

Step 3 — Build/train ML model

If you use multiple metrics in best subsets regression and each suggests a different “best” model, cross-validation error can give more insight into which set of variables most accurately predicts against your actual train/test data

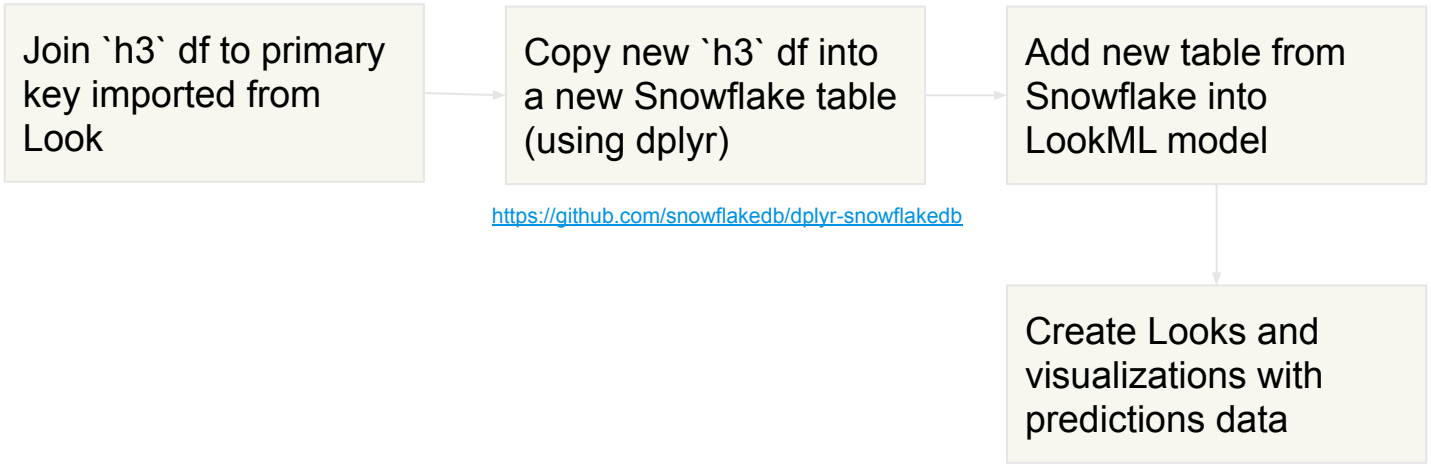
```
253  # Compute cross-validation error for HPI
254  model.ids <- 1:5
255  cv.errors <-  map(model.ids, get_model_formula, var_subset, "y.HPI") %>%
256    map(get_cv_error, data = model_x3) %>%
257    unlist()
258  cv.errors
259
260  # Select the model that minimize the CV error
261  which.min(cv.errors)
262
263  #Based on the response, select the best set of variables based on the number with lowest CV error
264  get_model_formula(5, var_subset, "y.HPI")
265
266  #This is the H3 model for predicting HPI
267  h3 <- lm(y.HPI ~ x.median_home_value + x.Three_yr_HPI_percent_change +
268       x.Ten_yr_HPI_percent_change + x.median_net_wealth + x.percent_of_households_renters, data
     =base_df2)
```

Dataframe `h3` contains our predictor values

looker

Step 4 — Make data available to Looker

Let's assume that we are using Snowflake as our data warehouse. Here is a simple process outlining how to make our predicted values available for modelling back within LookML.



The journey to data science with Looker

Is this right?

All data is stored on various disconnected Excel spreadsheets or databases

Are we tracking that?

Building clean data pipelines for Looker to model all data and be the single source of truth gives visibility to what is actually being tracked/measured and how

What does it really mean?

Simple statistical modeling helps create a picture of good vs bad conclusions to draw from data

What will happen tomorrow?

Strong modeling with minimized error allows for predictive analytics

looker

Questions?


looker


Thank you
Rate this session in
the JOIN mobile app
#JOINdata
@lookerdata