

Tableros Dinámicos de Visualización

Herramientas y Aplicaciones con Shiny

Curso de Visualización de Datos

2025-11-07

Introducción

Transformando datos estáticos en experiencias interactivas

Objetivos de la clase




Al finalizar esta sesión serán capaces de:

- **Comprender** qué son los tableros dinámicos y su importancia en el análisis moderno de datos
- **Identificar** las principales herramientas de Business Intelligence y sus casos de uso
- **Evaluar** ventajas y desventajas de PowerBI, Tableau, Looker Studio y Shiny
- **Entender** los fundamentos de Shiny para crear aplicaciones interactivas con R
- **Aplicar** conceptos de reactividad y deployment para compartir análisis

Estructura de la clase

Metodología: De lo conceptual a lo práctico

Para cada tema:

1. **Marco conceptual:** Qué son los tableros dinámicos y por qué importan
2. **Comparación de herramientas:** Ventajas, desventajas y casos de uso
3. **Foco en Shiny:**
 -  Por qué Shiny para análisis estadístico
 -  Consideraciones importantes
 -  Demo práctica
4. **Aplicación:** Explorando una aplicación Shiny real

Énfasis:

- Pensamiento crítico en selección de herramientas
- Interactividad para exploración de datos
- Reproducibilidad y control total con código

Bloque 1: Fundamentos

Del reporte estático al tablero dinámico

¿Qué son los Tableros Dinámicos?

Definición:

Los **tableros dinámicos** (dashboards) son interfaces visuales interactivas que permiten:

- Explorar **datos** en tiempo real
- Filtrar y **segmentar** información según necesidades
- Visualizar **múltiples métricas** simultáneamente
- Tomar **decisiones** basadas en datos actualizados
- Comunicar **insights** de forma efectiva

Impacto:

Son herramientas fundamentales en el análisis moderno de datos, permitiendo que usuarios no técnicos puedan explorar información compleja.

Del Reporte Estático al Tablero Dinámico

Reporte Estático Tradicional:

Características:

- PDF o Excel fijo
- Datos de un momento específico
- Requiere regeneración manual
- Limitado a lo que el analista previó

Ejemplo típico:

- Informe mensual en PDF
- Tablas y gráficos fijos
- Requiere nuevo informe para otras preguntas

Tablero Dinámico:

- Interfaz web interactiva
- Datos actualizados automáticamente
- El usuario puede explorar libremente
- Permite responder preguntas no anticipadas

Ejemplo típico:

- Dashboard actualizado en tiempo real
- Filtros interactivos
- Múltiples vistas de los mismos datos

Características Clave de los Tableros

1. Interactividad

- Filtros dinámicos para segmentar información
- Selección de variables para exploración
- Zoom y drill-down para profundizar

2. Actualización

- Datos en tiempo real o near-real-time
- Refrescamiento automático programado
- Conexión a fuentes de datos vivas

3. Visualización

- Gráficos múltiples y complementarios
- Indicadores clave (KPIs) destacados
- Tablas dinámicas y ordenables

Bloque 2: Herramientas de BI

Comparación de plataformas

Principales Herramientas de Business Intelligence

Herramienta	Tipo	Complejidad	Costo aproximado
PowerBI	Desktop/Cloud	Media	\$10-20/usuario/mes
Tableau	Desktop/Cloud	Media-Alta	\$15-70/usuario/mes
Looker Studio	Cloud	Baja	Gratuito
Shiny (R)	Código	Alta	Gratuito (open source)

Criterios de selección:

- Presupuesto disponible
- Complejidad del análisis requerido
- Habilidades técnicas del equipo
- Necesidad de compartir y colaborar
- Integración con sistemas existentes

PowerBI (Microsoft)



Ventajas:

- Integración perfecta con ecosistema Microsoft (Excel, Azure, SQL Server)
- Interfaz drag-and-drop muy intuitiva
- Potente motor de consultas (DAX) para cálculos complejos
- Versión gratuita disponible (PowerBI Desktop)
- Gran comunidad y documentación en español



Desventajas:

- Licencias costosas para compartir (PowerBI Pro/Premium)
- Menor flexibilidad en diseño personalizado
- Curva de aprendizaje pronunciada en DAX
- Dependiente de infraestructura Microsoft



Ideal para:

Empresas con infraestructura Microsoft, análisis de negocio tradicional, equipos con presupuesto para licencias

Tableau



Ventajas:

- Visualizaciones muy elegantes y pulidas
- Excelente para storytelling con datos
- Gran comunidad y recursos de aprendizaje
- Conexión a múltiples fuentes de datos
- Interfaz intuitiva para análisis exploratorio



Desventajas:

- Costo elevado (licencias entre \$15-\$70/usuario/mes)
- Puede ser lento con grandes volúmenes de datos
- Limitaciones en personalización avanzada
- Menos flexible para análisis estadístico complejo



Ideal para:

Presentaciones ejecutivas, análisis exploratorio visual sofisticado, empresas grandes con presupuesto

Looker Studio (Google)



Ventajas:

- Completamente gratuito
- Integración perfecta con Google (Analytics, Sheets, BigQuery)
- Colaboración en tiempo real
- No requiere instalación (100% cloud)
- Curva de aprendizaje suave



Desventajas:

- Funcionalidad limitada vs PowerBI/Tableau
- Personalización restringida
- Dependiente del ecosistema Google
- Menor capacidad para análisis estadístico avanzado



Ideal para:

Reportes web, marketing digital, equipos pequeños con presupuesto limitado, análisis de métricas web

Shiny (R/Python)



Ventajas:

- **Control total** sobre diseño y funcionalidad
- Integración completa con análisis estadístico (R/Python)
- Gratuito y open source
- **Reproducible y versionable** (todo es código)
- Ideal para métodos estadísticos complejos
- Sin límites de personalización



Desventajas:

- Requiere conocimientos de programación
- Deployment más complejo que plataformas cloud
- Mantenimiento técnico necesario
- Curva de aprendizaje más pronunciada



Ideal para:

Análisis estadístico avanzado, investigación académica, prototipado rápido, usuarios técnicos, cuando se requiere máxima flexibilidad

Comparación: ¿Cuál elegir?

Criterio	PowerBI	Tableau	Looker	Shiny
Facilidad de Uso	★ ★ ★ ★	★ ★ ★ ★	★ ★ ★ ★ ★	★ ★
Costo	★ ★ ★	★ ★	★ ★ ★ ★ ★	★ ★ ★ ★ ★
Flexibilidad	★ ★ ★	★ ★ ★ ★	★ ★	★ ★ ★ ★ ★
Análisis Estadístico	★ ★ ★	★ ★	★	★ ★ ★ ★ ★
Compartir	★ ★ ★ ★	★ ★ ★ ★	★ ★ ★ ★ ★	★ ★ ★
Personalización	★ ★ ★	★ ★ ★	★ ★	★ ★ ★ ★ ★

Escala: ★ (bajo) a ★ ★ ★ ★ ★ (alto)

Recomendación:

No hay "la mejor" herramienta - depende del contexto, presupuesto, habilidades y objetivos específicos

Bloque 3: Shiny en Detalle

Framework para aplicaciones web con R

¿Qué es Shiny?

Definición:

Shiny es un framework de R (también disponible para Python) que permite crear aplicaciones web interactivas directamente desde R, sin necesidad de conocer HTML, CSS o JavaScript (aunque se pueden usar para personalización avanzada).

Componentes básicos:

1. **UI (User Interface)**: Define la apariencia y los controles de entrada
2. **Server**: Contiene la lógica y procesamiento de datos
3. **Reactividad**: Sistema de actualización automática cuando cambian los inputs

```
library(shiny)
```

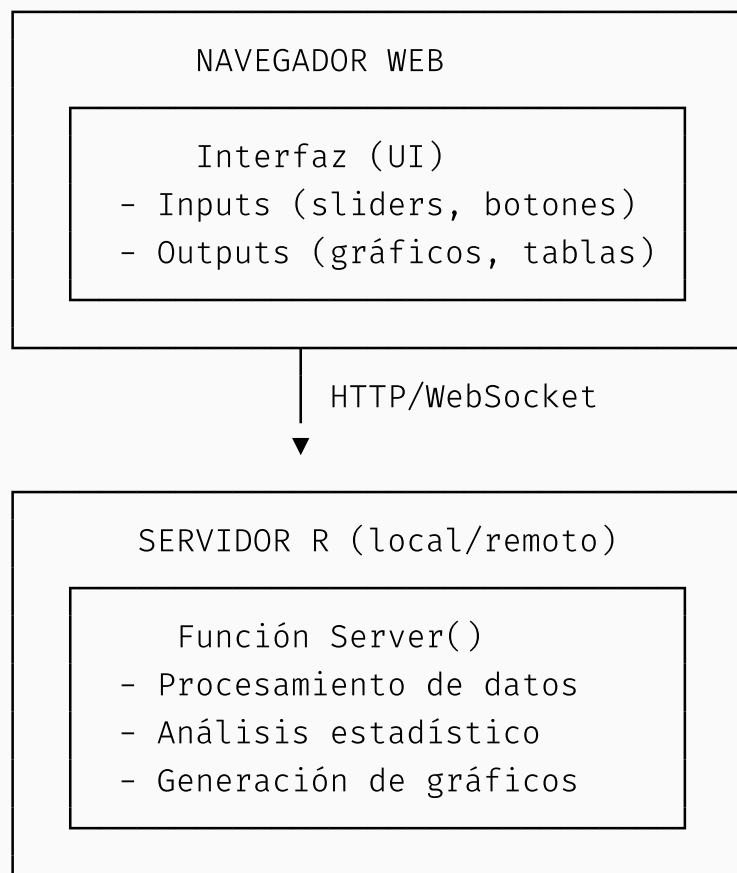
```
ui <- fluidPage(  
  # Interfaz del usuario  
)
```

```
server <- function(input, output) {  
  # Lógica del servidor  
}
```

```
shinyApp(ui = ui, server = server)
```

Arquitectura de Shiny

Flujo de información:



¿Ventajas frente a los otros?

Tipos de Inputs en Shiny

Widgets de entrada disponibles:

Básicos:

- `sliderInput()` - Deslizadores
- `selectInput()` - Menús desplegables
- `checkboxInput()` - Casillas individuales
- `radioButtons()` - Botones de radio
- `textInput()` - Campos de texto
- `numericInput()` - Inputs numéricos

Avanzados:

- `dateInput()` - Selectores de fecha
- `dateRangeInput()` - Rango de fechas
- `fileInput()` - Carga de archivos
- `actionButton()` - Botones de acción
- `checkboxGroupInput()` - Grupo de checks
- `sliderInput(range=TRUE)` - Rango de valores

Característica importante:

Todos los inputs están conectados reactivamente al servidor - cuando cambian, se actualizan los outputs automáticamente

Tipos de Outputs en Shiny

Formas de mostrar resultados:

Visualizaciones:

- `plotOutput()` / `renderPlot()` - Gráficos estáticos (ggplot2, base)
- `plotlyOutput()` / `renderPlotly()` - Gráficos interactivos
- `imageOutput()` / `renderImage()` - Imágenes

Tablas:

- `tableOutput()` / `renderTable()` - Tablas estáticas
- `dataTableOutput()` / `renderDataTable()` - Tablas interactivas (DT)

Patrón consistente:

Cada tipo de output tiene su función `xxxOutput()` en UI y `renderXxx()` en server

Texto y UI:

- `textOutput()` / `renderText()` - Texto simple
- `verbatimTextOutput()` / `renderPrint()` - Código/output
- `uiOutput()` / `renderUI()` - UI dinámica
- `htmlOutput()` / `renderUI()` - HTML personalizado

Layout en Shiny

Sistema de layouts disponibles:

Básicos:

- `fluidPage()` - Página responsive (se adapta al tamaño)
- `fluidRow()` y `column()` - Sistema de grillas (Bootstrap 12 columnas)
- `sidebarLayout()` - Panel lateral + panel principal (muy común)

Avanzados:

- `navbarPage()` - Navegación con pestañas superiores
- `tabsetPanel()` - Pestañas dentro de una sección
- `dashboardPage()` (shinydashboard) - Dashboards profesionales con menú lateral

Ejemplo de layout típico:

```
fluidPage(  
  titlePanel("Mi Dashboard"),  
  sidebarLayout(  
    sidebarPanel(width = 3,  
      # Controles aquí  
    )  
  )  
)
```

Reactividad: El Corazón de Shiny

¿Qué es la reactividad?

Es el mecanismo que actualiza automáticamente los outputs cuando cambian los inputs

```
server <- function(input, output) {  
  
  # 1. DATOS REACTIVOS  
  # Se recalculan cuando cambia input$dataset  
  datos <- reactive({  
    if (input$dataset == "mtcars") {  
      mtcars  
    } else {  
      iris  
    }  
  })  
  
  # 2. OUTPUT REACTIVO  
  # Se regenera cuando cambian los datos() o input$variable  
  output$grafico <- renderPlot({  
    ggplot(datos(), aes(x = .data[[input$variable]])) +  
      geom_histogram(bins = 30, fill = "steelblue")  
  })  
}
```

Ejemplo Completo: Mi Primera App Shiny

```
library(shiny)
library(ggplot2)

# INTERFAZ
ui <- fluidPage(
  titlePanel("Explorador de Datos"),

  sidebarLayout(
    sidebarPanel(
      selectInput("dataset", "Seleccionar dataset:",
                  choices = c("mtcars", "iris", "diamonds")),
      sliderInput("n", "Número de filas:",
                  min = 10, max = 100, value = 50)
    ),

    mainPanel(
      tabsetPanel(
        tabPanel("Tabla", tableOutput("tabla")),
        tabPanel("Resumen", verbatimTextOutput("resumen")),
        tabPanel("Gráfico", plotOutput("grafico"))
      )
    )
  )
)
```

Ventajas de Shiny para Análisis Estadístico

✓ Exploración interactiva de datos:

- Cambiar variables sobre la marcha sin reescribir código
- Probar diferentes parámetros de modelos
- Visualizar resultados en tiempo real
- Iteración rápida en el análisis

✓ Reproducibilidad:






- Todo el análisis está en código fuente
- Versionable con Git/GitHub
- Documentado y auditable
- Fácil de compartir con colaboradores

✓ Comunicación efectiva:

- Audiencias no técnicas pueden explorar por sí mismas
- Presentaciones interactivas más engaging
- Reportes dinámicos actualizables
- Permite responder preguntas ad-hoc

Shiny: Ejecución Local (On Premise)

Ventajas del despliegue local:

-  No requiere servidor externo ni hosting
-  Datos sensibles permanecen en tu computadora
-  Sin costos de hosting o licencias
-  Ideal para prototipos y desarrollo
-  Control total sobre el ambiente

Cómo ejecutar localmente:

Opción 1: Desde RStudio

Abrir el archivo app.R o ui.R/server.R

Presionar el botón "Run App"

Opción 2: Desde consola

```
shiny::runApp("ruta/a/tu/app")
```

Opción 3: Con parámetros específicos

```
shiny::runApp("ruta/a/tu/app",  
              port = 3838,          # Puerto personalizado  
              launch.browser = TRUE) # Abrir navegador automáticamente
```





Bloque 5: Deployment y Extensiones

Compartiendo tu trabajo





Deployment: Compartir tu Tablero

Opciones para publicar tu aplicación Shiny:




1. shinyapps.io (Posit Cloud)

-  Gratuito con límites (5 apps, 25 horas/mes)
-  Deployment super fácil desde RStudio
-  `rsconnect :: deployApp()`
-  Límites estrictos en plan gratuito

2. Shiny Server (Open Source)

-  Servidor propio (Linux)
-  Sin límites de uso
-  Requiere administración de servidor
-  Instalación y configuración más compleja

3. Docker Containers

-  Portable y reproducible
-  Escalable y versionable
-  Requiere conocimientos de Docker

Extendiendo el Tablero

Mejoras posibles para tus aplicaciones:

Más análisis estadísticos:

- Series temporales (ARIMA, STL decomposition)
- Modelos Generalizados Lineales (GLM)
- Árboles de decisión y random forests
- Análisis de supervivencia

Funcionalidades adicionales:

- Carga de datos personalizados (CSV, Excel, SPSS)
- Exportar resultados (download de gráficos, reportes PDF)
- Guardar configuraciones del usuario
- Integración con bases de datos

Visualizaciones avanzadas:

- Gráficos de redes con visNetwork
- Mapas interactivos con leaflet
- Visualizaciones 3D con plotly

Recursos para Aprender Shiny

Documentación oficial:

- shiny.posit.co - Tutorial oficial y guías
- [Shiny Gallery](#) - Ejemplos con código

Libros recomendados:

- *Mastering Shiny* by Hadley Wickham (gratuito online)
- *Engineering Production-Grade Shiny Apps* (libro avanzado)
- *Outstanding User Interfaces with Shiny* (interfaces avanzadas)

Paquetes complementarios esenciales:

- `shinydashboard` - Dashboards elegantes tipo admin panel
- `shinyWidgets` - Inputs adicionales y mejorados
- `shinyjs` - Control de JavaScript desde R
- `bs4Dash` - Templates Bootstrap 4
- `bslib` - Sistema de temas moderno
- `DT` - Tablas interactivas potentes
- `plotly` - Gráficos interactivos

Conclusiones

Los tableros dinámicos transforman el análisis de datos:

- ✓ **Interactividad** permite exploración profunda
- ✓ **Actualización automática** mantiene información al día
- ✓ **Múltiples herramientas** disponibles según necesidades
- ✓ **Shiny destaca** en análisis estadístico y reproducibilidad
- ✓ **Reactividad** hace los dashboards intuitivos
- ✓ **Código** permite flexibilidad total

