Turn your spreadsheets into intelligent data assistants that explain, analyze, and visualize your data automatically.

Managing data inside Google Sheets is something almost everyone does—students, analysts, project managers, educators, accountants, startup founders, and even executives. But interpreting that data often requires real time, skill, and patience.

What if Google Sheets could *explain your data to you*?

What if it could *find patterns, trends, anomalies, suggested formulas, and even recommended charts*—all automatically?

Thanks to **Google Apps Script** + **Gemini**, this is now possible.

Today, I'm excited to share a complete, open-source **Google Sheets AI Data Analyst**, packaged as an Apps Script extension that runs entirely inside your spreadsheet. No external servers. No database. No complicated setup.

Just open a sheet → select a range → click "Analyze."

⭐ What This Tool Can Do

Once installed, the AI Data Analyst adds a new menu inside Sheets:

AI Data Analyst → Open AI Analyst Panel

From the sidebar, you can choose several analysis types:

1. Explain this data

A plain-English description of what your selected data represents.

2. Find trends & patterns

- Time-based trends
- Categorical patterns
- Notable correlations

3. Detect anomalies & outliers

- Suspicious values
- Spikes or dips
- Misaligned categories

4. Suggest pivot tables

Concrete recommended configurations:

- Rows
- Columns
- Values
- Aggregations (SUM, AVG, COUNT)

5. Suggest formulas

- A1-style references
- Computed columns
- Cleanup formulas
- Aggregate formulas

6. Suggest charts

- Chart type
- X/Y axes
- Series

- Visual insights

Everything is powered by **gemini-2.0-flash**, one of Google's fastest reasoning models.

🧠 How It Works

The extension works entirely inside Google Sheets using Apps Script:

1. You select a range
2. The tool extracts the values as a **tab-separated table**
3. It sends a structured prompt to Gemini
4. Gemini analyzes the dataset in context
5. The sidebar displays the analysis with headings, bullets, and explanations

No data leaves the Google Apps Script environment except what you choose to send to Gemini. This makes it lightweight, private, and safe for business use.

📦 Download the Full Source Code

You can download the complete GitHub-ready ZIP here:
👉 **https://github.com/lsvekis/Google-Sheets-AI-Data-Analyst**

Includes:

- `Code.gs`
- `SidebarAIAnalyst.html`
- `appsscript.json`
- `README.md`
- `.gitignore`

You can upload this directly to GitHub, or paste the contents into Apps Script.

🛠️ Getting Started (Setup Guide)

1. Open Apps Script

In your Google Sheet:
Extensions → Apps Script

2. Add project files

Upload or paste:

- `Code.gs`
- `SidebarAIAnalyst.html`
- `appsscript.json`

3. Add your Gemini API key

In Apps Script:
Project Settings → Script Properties → Add Script Property

```
Key: GEMINI_API_KEY
Value: your_api_key_here
```

4. Reload your Sheet

You'll see the new menu:
AI Data Analyst → Open AI Analyst Panel
Select a range → choose an analysis type → click **Analyze**.

🔍 Example: Asking the AI to Find Trends

Suppose you select a table like:

Month	Sales
Jan	1200
Feb	1500
Mar	2400
Apr	1300

Ask:
"Find trends and patterns."
You'll get something like:

Trend Summary
• Sales rise sharply from January → March (+100% growth).
• April shows a significant dip, falling below the February level.
Possible Explanations
• Seasonal demand.
• Launch campaign in March.
Recommendations
• Investigate March's driver.
• Adjust April marketing strategy.

Clear, concise, and immediately useful.

🔮 Ideas for Future Enhancements

This is just the beginning.
Here are some powerful extensions you could build next:

- Auto-generate charts directly inside Sheets
- Auto-insert suggested formulas as new columns
- Save a history log of all analyses
- Generate dashboards based on data
- Integrate with BigQuery or Supabase
- Support CSV or Google Drive file imports
- Add one-click "Management Summary" mode

If you want help building any of these—just ask.

Code.gs – Apps Script backend

Create a new Apps Script project bound to a Google Sheet (Extensions → Apps Script) and paste this into `Code.gs`:

```
/**
 * Google Sheets AI Data Analyst (Apps Script + Gemini)
 *
 * - Adds "AI Data Analyst" menu to Google Sheets.
 * - Opens a sidebar that uses the current selection as data context.
 * - Gemini can:
 *    - Explain the data
 *    - Find trends
 *    - Detect anomalies
 *    - Suggest pivot summaries
 *    - Suggest formulas
 *    - Suggest charts
 */

const GEMINI_MODEL_ID = 'gemini-2.0-flash';
const MAX_DATA_CHARS = 8000; // Limit data size sent to Gemini

/**
 * Add custom menu when the spreadsheet opens.
 */
function onOpen() {
  SpreadsheetApp.getUi()
    .createMenu('AI Data Analyst')
    .addItem('Open AI Analyst Panel', 'showAiAnalystSidebar')
    .addToUi();
}

/**
 * Show the sidebar UI.
 */
function showAiAnalystSidebar() {
  const html =
HtmlService.createHtmlOutputFromFile('SidebarAIAnalyst')
    .setTitle('AI Data Analyst');
  SpreadsheetApp.getUi().showSidebar(html);
}

/**
 * Get Gemini API key from Script Properties.
 */
function getGeminiApiKey_() {
  const props = PropertiesService.getScriptProperties();
  const key = props.getProperty('GEMINI_API_KEY');
  if (!key) {
```

```
    throw new Error('GEMINI_API_KEY is not set in Script
Properties.');
  }
  return key;
}

/**
 * Read the currently selected range and build a textual
representation.
 */
function getSelectedRangeSnapshot() {
  const sheet = SpreadsheetApp.getActiveSheet();
  const range = sheet.getActiveRange();
  if (!range) {
    throw new Error('No range selected. Please select some data
first.');
  }

  const values = range.getDisplayValues();
  const numRows = values.length;
  const numCols = values[0].length;

  // Build a lightweight table representation (tab-separated)
  let tableText = '';
  for (let r = 0; r < numRows; r++) {
    const row = values[r].map(v => (v === '' ? '' : String(v)));
    tableText += row.join('\t') + '\n';
  }

  if (tableText.length > MAX_DATA_CHARS) {
    tableText = tableText.slice(0, MAX_DATA_CHARS) +
'\n...[truncated]';
  }

  return {
    sheetName: sheet.getName(),
    address: range.getA1Notation(),
    numRows,
    numCols,
    tableText
  };
}

/**
```

```
 * Main entry from the sidebar: analyze the selected range with the
chosen action.
 *
 * @param {string} action One of: explain, trends, anomalies, pivots,
formulas, charts
 * @param {string} extra Optional extra instructions
 * @returns {{analysisText: string}}
 */
function analyzeSelectedRange(action, extra) {
  const snapshot = getSelectedRangeSnapshot();
  const dataText = snapshot.tableText;

  const actionLabel = {
    explain: 'Explain this data',
    trends: 'Find trends and patterns',
    anomalies: 'Detect anomalies and outliers',
    pivots: 'Suggest pivot table summaries',
    formulas: 'Suggest useful formulas and calculated columns',
    charts: 'Suggest charts and visualizations'
  }[action] || 'Explain this data';

  const extraInstructions = extra ? extra.trim() : '';

  const systemInstruction = [
    'You are an experienced data analyst helping a user understand and
work with a small dataset from a spreadsheet.',
    'The data is provided in a tab-separated table format, including
headers when present.',
    '',
    'You MUST:',
    '- Interpret the data based only on what is visible.',
    '- Be explicit, concrete, and use clear language.',
    '- Whenever useful, reference column names directly.',
    '',
    'Depending on the requested action, you should:',
    '1) If the action is "Explain this data":',
    '   - Describe what the dataset appears to represent.',
    '   - Highlight any obvious patterns or structure.',
    '',
    '2) If the action is "Find trends and patterns":',
    '   - Describe trends over time, by category, or by value.',
    '   - Mention any correlations or strong relationships you can
infer.',
    '',
```

```
    '3) If the action is "Detect anomalies and outliers":',
    '   - Point out values or rows that look unusual.',
    '   - Explain why they might be anomalies.',
    '',
    '4) If the action is "Suggest pivot table summaries":',
    '   - Suggest concrete pivot configurations:',
    '     - Rows: <column>',
    '     - Columns: <column>',
    '     - Values: <column, aggregation>',
    '',
    '5) If the action is "Suggest formulas and calculated columns":',
    '   - Propose specific formulas (with A1-style references).',
    '   - Explain what each formula would calculate.',
    '',
    '6) If the action is "Suggest charts and visualizations":',
    '   - Recommend chart types (e.g., line, bar, scatter, pie).',
    '   - Specify which columns belong on each axis.',
    '',
    'ALWAYS respond in a structured, readable way with headings and
bullet points where helpful.'
  ].join('\n');

  const prompt =
    systemInstruction +
    '\n\nRequested analysis action: ' +
    actionLabel +
    (extraInstructions ? '\nExtra user instructions: ' +
extraInstructions : '') +
    '\n\nSpreadsheet context:\n' +
    '- Sheet name: ' +
    snapshot.sheetName +
    '\n- Range: ' +
    snapshot.address +
    '\n- Size: ' +
    snapshot.numRows +
    ' rows x ' +
    snapshot.numCols +
    ' cols' +
    '\n\nTAB-SEPARATED DATA START\n' +
    dataText +
    'TAB-SEPARATED DATA END\n';

  const apiKey = getGeminiApiKey_();
  const url =
```

```
    'https://generativelanguage.googleapis.com/v1beta/models/' +
    GEMINI_MODEL_ID +
    ':generateContent';

  const payload = {
    contents: [
      {
        role: 'user',
        parts: [{ text: prompt }]
      }
    ],
    generationConfig: {
      temperature: 0.4,
      maxOutputTokens: 768
    }
  };

  const response = UrlFetchApp.fetch(url, {
    method: 'post',
    muteHttpExceptions: true,
    contentType: 'application/json',
    headers: { 'x-goog-api-key': apiKey },
    payload: JSON.stringify(payload)
  });

  const httpCode = response.getResponseCode();
  const raw = response.getContentText();

  if (httpCode !== 200) {
    throw new Error('Gemini API error (HTTP ' + httpCode + '): ' +
raw);
  }

  let data;
  try {
    data = JSON.parse(raw);
  } catch (err) {
    throw new Error('Failed to parse Gemini response: ' + err + ' |
Raw: ' + raw);
  }

  if (!data.candidates || !data.candidates.length) {
    throw new Error('No candidates returned from Gemini. Raw: ' +
raw);
```

```
  }

  const parts =
    data.candidates[0].content && data.candidates[0].content.parts
      ? data.candidates[0].content.parts
      : [];

  const analysisText = parts
    .map(function (p) {
      return p.text || '';
    })
    .join('')
    .trim();

  return { analysisText: analysisText };
}
```

`SidebarAIAnalyst.html` – Sidebar UI

Add a new HTML file in Apps Script named `SidebarAIAnalyst` and paste:

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_top" />
    <meta charset="UTF-8" />
    <style>
      body {
        font-family: system-ui, -apple-system, BlinkMacSystemFont,
'Segoe UI',
          sans-serif;
        margin: 0;
        padding: 12px;
        background: #020617;
        color: #e5e7eb;
      }
      h1 {
        font-size: 1.1rem;
        margin: 0 0 4px;
      }
      .subtitle {
        font-size: 0.8rem;
        color: #9ca3af;
        margin-bottom: 10px;
      }
```

```
label {
  font-size: 0.8rem;
  display: block;
  margin-bottom: 4px;
}
select,
textarea,
button {
  font-family: inherit;
  font-size: 0.85rem;
}
select {
  width: 100%;
  padding: 6px 8px;
  border-radius: 6px;
  border: 1px solid #374151;
  background: #020617;
  color: #e5e7eb;
  margin-bottom: 8px;
}
textarea {
  width: 100%;
  min-height: 60px;
  padding: 6px 8px;
  border-radius: 6px;
  border: 1px solid #374151;
  background: #020617;
  color: #e5e7eb;
  resize: vertical;
  margin-bottom: 8px;
}
button {
  width: 100%;
  padding: 8px;
  border-radius: 6px;
  border: none;
  background: #3b82f6;
  color: white;
  font-weight: 600;
  cursor: pointer;
}
button:disabled {
  opacity: 0.6;
  cursor: default;
```

```
      }
      .status {
        font-size: 0.75rem;
        color: #9ca3af;
        margin: 6px 0;
        min-height: 1em;
      }
      .status.error {
        color: #f97373;
      }
      .output {
        margin-top: 8px;
        padding: 8px;
        border-radius: 8px;
        border: 1px solid #1f2937;
        background: #020617;
        font-size: 0.8rem;
        white-space: pre-wrap;
        max-height: 360px;
        overflow-y: auto;
      }
    </style>
  </head>
  <body>
    <h1>AI Data Analyst</h1>
    <div class="subtitle">
      Select a range in the sheet, then choose an analysis type and
click
      <strong>Analyze</strong>.
    </div>

    <label for="action">Analysis Type</label>
    <select id="action">
      <option value="explain">Explain this data</option>
      <option value="trends">Find trends & patterns</option>
      <option value="anomalies">Detect anomalies</option>
      <option value="pivots">Suggest pivot summaries</option>
      <option value="formulas">Suggest formulas</option>
      <option value="charts">Suggest charts</option>
    </select>

    <label for="extra">Extra instructions (optional)</label>
    <textarea
      id="extra"
```

```
      placeholder="E.g. Focus on revenue, keep it simple, assume
management audience..."
    ></textarea>

    <button id="runBtn" onclick="runAnalysis()">Analyze Selected
Range</button>

    <div id="status" class="status"></div>

    <div id="output" class="output"></div>

    <script>
      const statusEl = document.getElementById('status');
      const outputEl = document.getElementById('output');
      const actionEl = document.getElementById('action');
      const extraEl = document.getElementById('extra');
      const runBtn = document.getElementById('runBtn');

      function setLoading(isLoading, msg) {
        runBtn.disabled = isLoading;
        runBtn.textContent = isLoading ? 'Analyzing…' : 'Analyze
Selected Range';
        statusEl.textContent = msg || '';
        statusEl.classList.remove('error');
      }

      function showError(msg) {
        statusEl.textContent = msg;
        statusEl.classList.add('error');
      }

      function runAnalysis() {
        const action = actionEl.value;
        const extra = extraEl.value;

        setLoading(true, 'Sending data to Gemini…');

        google.script.run
          .withSuccessHandler(function (res) {
            setLoading(false, 'Analysis complete.');
            outputEl.textContent = res.analysisText || '(No analysis
returned.)';
          })
          .withFailureHandler(function (err) {
```

```
            setLoading(false, '');
            showError(err.message || String(err));
          })
          .analyzeSelectedRange(action, extra);
      }
    </script>
  </body>
</html>
```

Configure the API key

In Apps Script:

1. Go to **Project Settings**
2. Under **Script properties**, add:
    - **Key:** `GEMINI_API_KEY`
    - **Value:** your Gemini API key

How to use it

1. Open your Sheet.
2. Select a data range (include headers if you have them).
3. Go to **AI Data Analyst → Open AI Analyst Panel**.
4. In the sidebar:
    - Choose **Analysis Type** (e.g. “Find trends & patterns”).
    - Optionally add extra instructions.
    - Click **Analyze Selected Range**.

You’ll get:

- Plain-English explanations
- Trends & patterns
- Anomaly descriptions
- Suggested pivot configurations
- Example formulas with A1 references
- Recommended chart types + which columns to use

💬 Final Thoughts

AI is changing how we work with data—but most tools require exporting your files, using cloud dashboards, or writing Python. This project shows that **you can bring real data intelligence directly into Google Sheets**, using nothing but Apps Script and Gemini.

It’s lightweight.
It’s fast.
It’s private.
And it unlocks an entirely new way to work with spreadsheets.