Account Anomaly Detector



Account Anomaly Detector alerts the advertiser whenever an AdWords account is suddenly behaving too differently from what's historically observed. When an issue is encountered, the script will send the user an alerting email. Only a single email for an alert is sent per day.

The script is comparing stats observed so far today with historical stats for the same day of week. For instance, stats for a Tuesday, 13:00 are compared with stats for 26 previous Tuesdays. Adjust the number of weeks to look back depending on the age and stability of your account.



Scheduling

Schedule the script to run **Hourly** in order to get the most out of alerting. If the alert is too noisy, scheduling it **Daily** around mid-day might also make sense.

How it works

Suppose the script runs at 7pm on a Tuesday. Since AdWords statistics may be [up to 3 hours delayed](https://support.google.com/adwords/answer/2544985), the script will only consider stats up to 4pm.

The script will then fetch stats for 26 preceding Tuesdays, average them, and compare with today's stats.

No subsequent alerts of the same type will be triggered for the day. If you'd like to reset the alert, delete the **Alerting** cell value.

Setup

* Setup a spreadsheet-based script with the source code below. Use Account Anomaly Detector [template spreadsheet](http://goo.gl/wQcKOX).
* Don't forget to update SPREADSHEET\_URL in code.
* Schedule the script **Hourly**.

[Setting up a spreadsheet-based script](https://developers.google.com/adwords/scripts/docs/solutions/account-anomaly-detector)

Source code

var SPREADSHEET\_URL = "[YOUR\_URL]";

var DAYS = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday'];

function main() {
  var spreadsheet = SpreadsheetApp.openByUrl(SPREADSHEET\_URL);
  spreadsheet.getRangeByName("date").setValue(new Date());
  spreadsheet.getRangeByName("account\_id").setValue(AdWordsApp.currentAccount().getCustomerId());
  var impressionsThreshold = parseField(spreadsheet.getRangeByName("impressions").getValue());
  var clicksThreshold = parseField(spreadsheet.getRangeByName("clicks").getValue());
  var costThreshold = parseField(spreadsheet.getRangeByName("cost").getValue());
  var weeksStr = spreadsheet.getRangeByName("weeks").getValue();
  var weeks = parseInt(weeksStr.substring(0, weeksStr.indexOf(" ")));
  var email = spreadsheet.getRangeByName("email").getValue();

  var now = new Date(Utilities.formatDate(new Date(),
      AdWordsApp.currentAccount().getTimeZone(), "MMM dd,yyyy HH:mm:ss"));

  var currentDate = now.getDate();
  now.setTime(now.getTime() - 3 \* 3600 \* 1000);
  var adjustedDate = now.getDate();

  var hours = now.getHours();
  if (hours == 0) {
    hours = 24;
  }
  if (hours == 1) {
    // first run of the day, kill existing alerts
    spreadsheet.getRangeByName("clicks\_alert").clearContent();
    spreadsheet.getRangeByName("impressions\_alert").clearContent();
    spreadsheet.getRangeByName("cost\_alert").clearContent();
  }
  var dayToCheck;
  if (currentDate != adjustedDate) {
    dayToCheck = 1;
  } else {
    dayToCheck = 0;
  }
  var dateRangeToCheck = getDateInThePast(dayToCheck);
  var dateRangeToEnd = getDateInThePast(dayToCheck + 1);
  var dateRangeToStart = getDateInThePast(dayToCheck + 1 + weeks \* 7);
  var fields = "HourOfDay,DayOfWeek,Clicks,Impressions,Cost";

  var today = AdWordsApp.report("SELECT " + fields +
      " FROM ACCOUNT\_PERFORMANCE\_REPORT DURING " + dateRangeToCheck + "," + dateRangeToCheck);
  var past = AdWordsApp.report("SELECT " + fields +
      " FROM ACCOUNT\_PERFORMANCE\_REPORT WHERE DayOfWeek=" + DAYS[now.getDay()].toUpperCase() +
      " DURING " + dateRangeToStart + "," + dateRangeToEnd);

  var todayStats = accumulateRows(today.rows(), hours, 1);
  var pastStats = accumulateRows(past.rows(), hours, weeks);

  var alertText = [];
  if (impressionsThreshold && todayStats.impressions < pastStats.impressions \* impressionsThreshold) {
    var range = spreadsheet.getRangeByName("impressions\_alert");
    if (!range.getValue() || range.getValue().length == 0) {
      alertText.push("    Impressions are too low: " + todayStats.impressions +
          " impressions by " + hours + ":00, expecting at least " +
          parseInt(pastStats.impressions \* impressionsThreshold));
      range.setValue("Alerting " + hours + ":00");
    }
  }
  if (clicksThreshold && todayStats.clicks < pastStats.clicks \* clicksThreshold) {
    var range = spreadsheet.getRangeByName("clicks\_alert");
    if (!range.getValue() || range.getValue().length == 0) {
      alertText.push("    Clicks are too low: " + todayStats.clicks +
          " clicks by " + hours + ":00, expecting at least " +
          (pastStats.clicks \* clicksThreshold).toFixed(1));
      range.setValue("Alerting " + hours + ":00");
    }
  }
  if (costThreshold && todayStats.cost > pastStats.cost \* costThreshold) {
    var range = spreadsheet.getRangeByName("clicks\_alert");
    if (!range.getValue() || range.getValue().length == 0) {
      alertText.push("    Cost is too high: " + todayStats.cost + " " +
          AdWordsApp.currentAccount().getCurrencyCode() + " by " + hours +
          ":00, expecting at most " + (pastStats.clicks \* costThreshold).toFixed(2));
      range.setValue("Alerting " + hours + ":00");
    }
  }
  if (alertText.length > 0 && email && email.length > 0) {
    MailApp.sendEmail(email,
        "AdWords Account " + AdWordsApp.currentAccount().getCustomerId() + " misbehaves",
        "Your account " + AdWordsApp.currentAccount().getCustomerId() +
        " is not performing as expected today: \n\n" + alertText.join("\n") +
        "\n\nLog into AdWords and take a look.\n\nAlerts dashboard: " + SPREADSHEET\_URL);
  }
  spreadsheet.getRangeByName("date").setValue(new Date());
  spreadsheet.getRangeByName("account\_id").setValue(AdWordsApp.currentAccount().getCustomerId());
  spreadsheet.getRangeByName("timestamp").setValue(DAYS[now.getDay()] + ", " + hours + ":00");

  var dataRows = [
    [todayStats.impressions, pastStats.impressions.toFixed(0)],
    [todayStats.clicks, pastStats.clicks.toFixed(1)],
    [todayStats.cost, pastStats.cost.toFixed(2)]
  ];
  spreadsheet.getRangeByName("data").setValue(dataRows);
}

function parseField(value) {
  if (value == "No alert") {
    return null;
  } else {
    return parseFloat(value);
  }
}

function accumulateRows(rows, hours, weeks) {
  var row;
  var result = null;

  for (var i = 0; i < hours; i ++) {
    if (rows.hasNext()) {
      row = rows.next();
    } else {
      row = null;
    }
    if (row && row ['HourOfDay'] == i) {
      result = addRow(row, result, 1 / weeks);
    } else {
      result = addRow(null, result, 1 / weeks);
    }
  }

  return result;
}

function addRow(row, previous, coefficient) {
  if (!coefficient) {
    coefficient = 1;
  }
  if (row == null) {
    row = {Clicks: 0, Impressions: 0, Cost: 0};
  }
  if (!previous) {
    return {
      clicks: parseInt(row['Clicks']) \* coefficient,
      impressions: parseInt(row['Impressions']) \* coefficient,
      cost: parseFloat(row['Cost']) \* coefficient,
    };
  } else {
    return {
      clicks: parseInt(row['Clicks']) \* coefficient + previous.clicks,
      impressions: parseInt(row['Impressions']) \* coefficient + previous.impressions,
      cost: parseFloat(row['Cost']) \* coefficient + previous.cost,
    };
  }
}

function checkInRange(today, yesterday, coefficient, field) {
  var yesterdayValue = yesterday[field] \* coefficient;
  if (today[field] > yesterdayValue \* 2) {
    Logger.log("" + field + " too much");
  } else if (today[field] < yesterdayValue / 2) {
    Logger.log("" + field + " too little");
  }
}

// Returns YYYYMMDD-formatted date.
function getDateInThePast(numDays) {
  var today = new Date();
  today.setDate(today.getDate() - numDays);
  return Utilities.formatDate(today, "PST", "yyyyMMdd");
}