```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": [],
      "authorship_tag": "ABX9TyOA8TUbzWt+57MSJxJef0yp",
      "include_colab_link": true
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "view-in-github",
        "colab_type": "text"
      },
      "source": [
        "<a
href=\"https://colab.research.google.com/github/Basha1605/python-mini-project/blob/main/Bash
a_dataset.ipynb\" target=\"_parent\"><img
src=\"https://colab.research.google.com/assets/colab-badge.svg\" alt=\"Open In Colab\"/></a>"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": null,
      "metadata": {
        "id": "BIIQ-En729fD"
      },
      "outputs": [],
      "source": [
        "#read a CSV file\n",
        "from google.colab import files"
      ]
    },
    {
```

```
    "cell_type": "code",
    "source": [
      "uploaded = files.upload()"
    ],
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/",
        "height": 128
      },
      "id": "Dk_dUxVw4CgR",
      "outputId": "46a0ab41-bbea-4349-f561-6c685626b861"
    },
    "execution_count": null,
    "outputs": [
      {
        "output_type": "display_data",
        "data": {
          "text/plain": [
            "<IPython.core.display.HTML object>"
          ],
          "text/html": [
            "\n",
            "     <input type=\"file\" id=\"files-94e838a4-a385-4c51-bb65-6bf02ae33ae4\"
name=\"files[]\" multiple disabled\n",
            "        style=\"border:none\" />\n",
            "     <output id=\"result-94e838a4-a385-4c51-bb65-6bf02ae33ae4\">\n",
            "      Upload widget is only available when the cell has been executed in the\n",
            "      current browser session. Please rerun this cell to enable.\n",
            "      </output>\n",
            "      <script>// Copyright 2017 Google LLC\n",
            "//\n",
            "// Licensed under the Apache License, Version 2.0 (the \"License\");\n",
            "// you may not use this file except in compliance with the License.\n",
            "// You may obtain a copy of the License at\n",
            "//\n",
            "//      http://www.apache.org/licenses/LICENSE-2.0\n",
            "//\n",
            "// Unless required by applicable law or agreed to in writing, software\n",
            "// distributed under the License is distributed on an \"AS IS\" BASIS,\n",
            "// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.\n",
            "// See the License for the specific language governing permissions and\n",
            "// limitations under the License.\n",
            "\n",
```

```
"/**\n",
" * @fileoverview Helpers for google.colab Python module.\n",
" */\n",
"(function(scope) {\n",
"function span(text, styleAttributes = {}) {\n",
"  const element = document.createElement('span');\n",
"  element.textContent = text;\n",
"  for (const key of Object.keys(styleAttributes)) {\n",
"    element.style[key] = styleAttributes[key];\n",
"  }\n",
"  return element;\n",
"}\n",
"\n",
"// Max number of bytes which will be uploaded at a time.\n",
"const MAX_PAYLOAD_SIZE = 100 * 1024;\n",
"\n",
"function _uploadFiles(inputId, outputId) {\n",
"  const steps = uploadFilesStep(inputId, outputId);\n",
"  const outputElement = document.getElementById(outputId);\n",
"  // Cache steps on the outputElement to make it available for the next call\n",
"  // to uploadFilesContinue from Python.\n",
"  outputElement.steps = steps;\n",
"\n",
"  return _uploadFilesContinue(outputId);\n",
"}\n",
"\n",
"// This is roughly an async generator (not supported in the browser yet),\n",
"// where there are multiple asynchronous steps and the Python side is going\n",
"// to poll for completion of each step.\n",
"// This uses a Promise to block the python side on completion of each step,\n",
"// then passes the result of the previous step as the input to the next step.\n",
"function _uploadFilesContinue(outputId) {\n",
"  const outputElement = document.getElementById(outputId);\n",
"  const steps = outputElement.steps;\n",
"\n",
"  const next = steps.next(outputElement.lastPromiseValue);\n",
"  return Promise.resolve(next.value.promise).then((value) => {\n",
"    // Cache the last promise value to make it available to the next\n",
"    // step of the generator.\n",
"    outputElement.lastPromiseValue = value;\n",
"    return next.value.response;\n",
"  });\n",
"}\n",
"\n",
```

```
"/**\n",
" * Generator function which is called between each async step of the upload\n",
" * process.\n",
" * @param {string} inputId Element ID of the input file picker element.\n",
" * @param {string} outputId Element ID of the output display.\n",
" * @return {!Iterable<!Object>} Iterable of next steps.\n",
" */\n",
"function* uploadFilesStep(inputId, outputId) {\n",
"  const inputElement = document.getElementById(inputId);\n",
"  inputElement.disabled = false;\n",
"\n",
"  const outputElement = document.getElementById(outputId);\n",
"  outputElement.innerHTML = '';\n",
"\n",
"  const pickedPromise = new Promise((resolve) => {\n",
"    inputElement.addEventListener('change', (e) => {\n",
"      resolve(e.target.files);\n",
"    });\n",
"  });\n",
"\n",
"  const cancel = document.createElement('button');\n",
"  inputElement.parentElement.appendChild(cancel);\n",
"  cancel.textContent = 'Cancel upload';\n",
"  const cancelPromise = new Promise((resolve) => {\n",
"    cancel.onclick = () => {\n",
"      resolve(null);\n",
"    };\n",
"  });\n",
"\n",
"  // Wait for the user to pick the files.\n",
"  const files = yield {\n",
"    promise: Promise.race([pickedPromise, cancelPromise]),\n",
"    response: {\n",
"      action: 'starting',\n",
"    }\n",
"  };\n",
"\n",
"  cancel.remove();\n",
"\n",
"  // Disable the input element since further picks are not allowed.\n",
"  inputElement.disabled = true;\n",
"\n",
"  if (!files) {\n",
"    return {\n",
```

```
"      response: {\n",
"        action: 'complete',\n",
"      }\n",
"    };\n",
"  }\n",
"\n",
"  for (const file of files) {\n",
"    const li = document.createElement('li');\n",
"    li.append(span(file.name, {fontWeight: 'bold'}));\n",
"    li.append(span(\n",
"        `(${file.type || 'n/a'}) - ${file.size} bytes, ` +\n",
"        `last modified: ${\n",
"            file.lastModifiedDate ? file.lastModifiedDate.toLocaleDateString() :\n",
"                                    'n/a'} - `));\n",
"    const percent = span('0% done');\n",
"    li.appendChild(percent);\n",
"\n",
"    outputElement.appendChild(li);\n",
"\n",
"    const fileDataPromise = new Promise((resolve) => {\n",
"      const reader = new FileReader();\n",
"      reader.onload = (e) => {\n",
"        resolve(e.target.result);\n",
"      };\n",
"      reader.readAsArrayBuffer(file);\n",
"    });\n",
"    // Wait for the data to be ready.\n",
"    let fileData = yield {\n",
"      promise: fileDataPromise,\n",
"      response: {\n",
"        action: 'continue',\n",
"      }\n",
"    };\n",
"\n",
"    // Use a chunked sending to avoid message size limits. See b/62115660.\n",
"    let position = 0;\n",
"    do {\n",
"      const length = Math.min(fileData.byteLength - position, MAX_PAYLOAD_SIZE);\n",
"      const chunk = new Uint8Array(fileData, position, length);\n",
"      position += length;\n",
"\n",
"      const base64 = btoa(String.fromCharCode.apply(null, chunk));\n",
"      yield {\n",
"        response: {\n",
```

```
          "          action: 'append',\n",
          "          file: file.name,\n",
          "          data: base64,\n",
          "        },\n",
          "      };\n",
          "\n",
          "      let percentDone = fileData.byteLength === 0 ?\n",
          "          100 :\n",
          "          Math.round((position / fileData.byteLength) * 100);\n",
          "      percent.textContent = `${percentDone}% done`;\n",
          "\n",
          "    } while (position < fileData.byteLength);\n",
          "  }\n",
          "\n",
          "  // All done.\n",
          "  yield {\n",
          "    response: {\n",
          "      action: 'complete',\n",
          "    }\n",
          "  };\n",
          "}\n",
          "\n",
          "scope.google = scope.google || {};\n",
          "scope.google.colab = scope.google.colab || {};\n",
          "scope.google.colab._files = {\n",
          "  _uploadFiles,\n",
          "  _uploadFilesContinue,\n",
          "};\n",
          "})(self);\n",
          "</script> "
        ]
      },
      "metadata": {}
    },
    {
      "output_type": "stream",
      "name": "stdout",
      "text": [
        "Saving Credit_Card_Applications.csv to Credit_Card_Applications.csv\n"
      ]
    }
  ]
},
{
```

```
    "cell_type": "code",
    "source": [
      "uploaded"
    ],
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/"
      },
      "id": "gN1_Riwp5DvA",
      "outputId": "5c667010-240a-45a0-f2b3-35ed0afedbe5"
    },
    "execution_count": null,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "{'Credit_Card_Applications.csv':
b'CustomerID,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,Class\\r\\n15776156,1,22.0
8,11.46,2,4,4,1.585,0,0,0,1,2,100,1213,0\\r\\n15739548,0,22.67,7,2,8,4,0.165,0,0,0,0,2,160,1,0\\
r\\n15662854,0,29.58,1.75,1,4,4,1.25,0,0,0,1,2,280,1,0\\r\\n15687688,0,21.67,11.5,1,5,3,0,1,1,1
1,1,2,0,1,1\\r\\n15715750,1,20.17,8.17,2,6,4,1.96,1,1,14,0,2,60,159,1\\r\\n15571121,0,15.83,0.5
85,2,8,8,1.5,1,1,2,0,2,100,1,1\\r\\n15726466,1,17.42,6.5,2,3,4,0.125,0,0,0,0,2,60,101,0\\r\\n156
60390,0,58.67,4.46,2,11,8,3.04,1,1,6,0,2,43,561,1\\r\\n15663942,1,27.83,1,1,2,8,3,0,0,0,0,2,176
,538,0\\r\\n15638610,0,55.75,7.08,2,4,8,6.75,1,1,3,1,2,100,51,0\\r\\n15644446,1,33.5,1.75,2,14,
8,4.5,1,1,4,1,2,253,858,1\\r\\n15585892,1,41.42,5,2,11,8,5,1,1,6,1,2,470,1,1\\r\\n15609356,1,20.
67,1.25,1,8,8,1.375,1,1,3,1,2,140,211,0\\r\\n15803378,1,34.92,5,2,14,8,7.5,1,1,6,1,2,0,1001,1\\r\
\n15599440,1,58.58,2.71,2,8,4,2.415,0,0,0,1,2,320,1,0\\r\\n15692408,1,48.08,6.04,2,4,4,0.04,0,
0,0,0,2,0,2691,1\\r\\n15683168,1,29.58,4.5,2,9,4,7.5,1,1,2,1,2,330,1,1\\r\\n15790254,0,18.92,9,
2,6,4,0.75,1,1,2,0,2,88,592,1\\r\\n15767729,1,20,1.25,1,4,4,0.125,0,0,0,0,2,140,5,0\\r\\n157686
00,0,22.42,5.665,2,11,4,2.585,1,1,7,0,2,129,3258,1\\r\\n15699839,0,28.17,0.585,2,6,4,0.04,0,0,
0,0,2,260,1005,0\\r\\n15786237,0,19.17,0.585,1,6,4,0.585,1,0,0,1,2,160,1,0\\r\\n15694530,1,41.
17,1.335,2,2,4,0.165,0,0,0,0,2,168,1,0\\r\\n15796813,1,41.58,1.75,2,4,4,0.21,1,0,0,0,2,160,1,0\\
r\\n15605791,1,19.5,9.585,2,6,4,0.79,0,0,0,0,2,80,351,0\\r\\n15714087,1,32.75,1.5,2,13,8,5.5,1,
1,3,1,2,0,1,1\\r\\n15711446,1,22.5,0.125,1,4,4,0.125,0,0,0,0,2,200,71,0\\r\\n15588123,1,33.17,3.
04,1,8,8,2.04,1,1,1,1,2,180,18028,1\\r\\n15748552,0,30.67,12,2,8,4,2,1,1,1,0,2,220,20,1\\r\\n15
618410,1,23.08,2.5,2,8,4,1.085,1,1,11,1,2,60,2185,1\\r\\n15672432,1,27,0.75,2,8,8,4.25,1,1,3,1,
2,312,151,1\\r\\n15610042,0,20.42,10.5,1,14,8,0,0,0,0,1,2,154,33,0\\r\\n15580914,1,52.33,1.37
5,1,8,8,9.46,1,0,0,1,2,200,101,0\\r\\n15583680,1,23.08,11.5,2,9,8,2.125,1,1,11,1,2,290,285,1\\r\\
n15813718,1,42.83,1.25,2,7,4,13.875,0,1,1,1,2,352,113,0\\r\\n15767264,1,74.83,19,1,1,1,0.04,0
,1,2,0,2,0,352,0\\r\\n15686461,1,25,12.5,2,6,4,3,1,0,0,1,1,20,1,1\\r\\n15678882,1,39.58,13.915,
2,9,4,8.625,1,1,6,1,2,70,1,1\\r\\n15789611,0,47.75,8,2,8,4,7.875,1,1,6,1,2,0,1261,1\\r\\n1566867
9,0,47.42,3,2,14,4,13.875,1,1,2,1,2,519,1705,1\\r\\n15631685,1,23.17,0,2,13,4,0.085,1,0,0,0,2,
0,1,1\\r\\n15655658,1,22.58,1.5,1,6,4,0.54,0,0,0,1,2,120,68,0\\r\\n15753591,1,26.75,1.125,2,14,
```

8,1.25,1,0,0,0,2,0,5299,1\\r\\n15617348,1,63.33,0.54,2,8,4,0.585,1,1,3,1,2,180,1,0\\r\\n1570458
1,1,23.75,0.415,1,8,4,0.04,0,1,2,0,2,128,7,0\\r\\n15738487,0,20.75,10.25,2,11,4,0.71,1,1,2,1,2,4
9,1,1\\r\\n15648069,0,24.5,1.75,1,8,4,0.165,0,0,0,0,2,132,1,0\\r\\n15737627,1,16.17,0.04,2,8,4,
0.04,0,0,0,0,2,0,1,1\\r\\n15731586,0,29.5,2,1,10,8,2,0,0,0,0,2,256,18,0\\r\\n15757467,0,52.83,1
5,2,8,4,5.5,1,1,14,0,2,0,2201,1\\r\\n15597709,1,32.33,3.5,2,4,4,0.5,0,0,0,1,2,232,1,0\\r\\n15720
529,1,21.08,4.125,1,3,8,0.04,0,0,0,0,2,140,101,0\\r\\n15596797,1,28.17,0.125,1,4,4,0.085,0,0,0
,0,2,216,2101,0\\r\\n15681755,1,19,1.75,1,8,4,2.335,0,0,0,1,2,112,7,0\\r\\n15815271,1,27.58,3.2
5,1,11,8,5.085,0,1,2,1,2,369,2,0\\r\\n15682860,1,27.83,1.5,2,9,4,2,1,1,11,1,2,434,36,1\\r\\n1562
1546,1,40,6.5,2,6,5,3.5,1,1,1,0,2,0,501,1\\r\\n15705918,0,37.33,2.5,2,3,8,0.21,0,0,0,0,2,260,24
7,0\\r\\n15684512,1,42.5,4.915,1,9,4,3.165,1,0,0,1,2,52,1443,1\\r\\n15671769,1,56.75,12.25,2,7
,4,1.25,1,1,4,1,2,200,1,1\\r\\n15642934,1,43.17,5,2,3,5,2.25,0,0,0,1,2,141,1,0\\r\\n15594305,0,2
3.75,0.71,2,9,4,0.25,0,1,1,1,2,240,5,0\\r\\n15789201,1,18.5,2,2,3,4,1.5,1,1,2,0,2,120,301,1\\r\\n
15706762,0,40.83,3.5,2,3,5,0.5,0,0,0,0,1,1160,1,0\\r\\n15766183,0,24.5,0.5,2,11,8,1.5,1,0,0,0,2,
280,825,1\\r\\n15777994,1,42,9.79,2,14,8,7.96,1,1,8,0,2,0,1,1\\r\\n15568162,0,19.5,0.165,2,11,4
,0.04,0,0,0,1,2,380,1,0\\r\\n15680643,1,21.5,11.5,2,3,4,0.5,1,0,0,1,2,100,69,0\\r\\n15761854,1,3
1.25,2.835,2,1,1,0,0,1,5,0,2,176,147,0\\r\\n15730793,1,27.25,1.585,2,13,8,1.835,1,1,12,1,2,583
,714,1\\r\\n15692137,1,48.75,26.335,1,1,1,0,1,0,0,1,2,0,1,0\\r\\n15608595,0,30.42,1.375,2,9,8,0
.04,0,1,3,0,2,0,34,0\\r\\n15709459,1,29.42,1.25,2,9,4,1.75,0,0,0,0,2,200,1,0\\r\\n15775750,1,28.
25,5.04,1,8,5,1.5,1,1,8,1,2,144,8,1\\r\\n15585855,1,40.25,21.5,2,10,9,20,1,1,11,0,2,0,1201,1\\r\\
n15752139,1,36.5,4.25,2,11,4,3.5,0,0,0,0,2,454,51,0\\r\\n15768295,1,25.58,0.335,2,4,8,3.5,0,0,
0,1,2,340,1,0\\r\\n15766906,1,29.83,3.5,2,8,4,0.165,0,0,0,0,2,216,1,0\\r\\n15725776,1,23.08,0,2
,4,4,1,0,1,11,0,1,0,1,0\\r\\n15682576,0,32.17,1.46,2,9,4,1.085,1,1,16,0,2,120,2080,1\\r\\n157040
81,1,25.17,3.5,2,13,4,0.625,1,1,7,0,2,0,7060,1\\r\\n15719940,0,35.17,3.75,2,1,1,0,0,1,6,0,2,0,2
01,0\\r\\n15672894,0,18.58,10,2,2,4,0.415,0,0,0,0,2,80,43,0\\r\\n15667451,1,39.92,5,2,3,5,0.21,
0,0,0,0,2,550,1,0\\r\\n15636767,1,23.42,1,2,8,4,0.5,0,0,0,1,1,280,1,0\\r\\n15571415,1,37.58,0,2,
8,4,0,0,0,0,0,3,184,1,1\\r\\n15575605,0,24.75,13.665,2,11,8,1.5,0,0,0,0,2,280,2,0\\r\\n15649160,
0,47,13,2,3,5,5.165,1,1,9,1,2,0,1,1\\r\\n15615832,1,34.17,5.25,2,9,4,0.085,0,0,0,1,2,290,7,1\\r\\
n15600975,1,22.17,0.585,1,1,1,0,0,0,0,0,2,100,1,0\\r\\n15690772,1,27.75,1.29,2,4,8,0.25,0,0,0,
1,1,140,1,0\\r\\n15565714,1,42.75,4.085,2,6,4,0.04,0,0,0,0,2,108,101,0\\r\\n15763108,1,28.67,1
4.5,2,2,4,0.125,0,0,0,0,2,0,287,0\\r\\n15723884,1,36.25,5,2,8,5,2.5,1,1,6,0,2,0,368,1\\r\\n15644
453,0,18.17,10,1,11,8,0.165,0,0,0,0,2,340,1,0\\r\\n15655464,1,21.25,1.5,2,9,4,1.5,0,0,0,0,2,150,
9,1\\r\\n15783883,0,38.92,1.665,2,6,4,0.25,0,0,0,0,2,0,391,0\\r\\n15787693,1,31.83,0.04,1,7,4,0
.04,0,0,0,0,2,0,1,0\\r\\n15664793,0,17.33,9.5,2,6,4,1.75,0,1,10,1,2,0,11,0\\r\\n15642391,0,20.42,
0.835,2,11,4,1.585,1,1,1,0,2,0,1,1\\r\\n15756538,0,39.08,4,2,8,4,3,0,0,0,0,2,480,1,0\\r\\n156688
30,1,38.67,0.21,2,4,4,0.085,1,0,0,1,2,280,1,1\\r\\n15796569,1,27.67,13.75,2,9,4,5.75,1,0,0,1,2,
487,501,1\\r\\n15677112,1,27.75,0.585,1,13,4,0.25,1,1,2,0,2,260,501,1\\r\\n15815040,1,19,0,1,1
,1,0,0,1,4,0,2,45,2,0\\r\\n15590434,0,25,0.875,2,14,8,1.04,1,0,0,1,2,160,5861,1\\r\\n15597536,1
,27.67,2,2,14,8,1,1,1,4,0,2,140,7545,1\\r\\n15723989,1,22.25,9,2,6,4,0.085,0,0,0,0,2,0,1,0\\r\\n1
5767358,0,49.83,13.585,2,4,8,8.5,1,0,0,1,2,0,1,1\\r\\n15594812,1,32.33,2.5,2,8,4,1.25,0,0,0,1,2
,280,1,0\\r\\n15688210,1,38.25,10.125,1,4,4,0.125,0,0,0,0,2,160,1,0\\r\\n15681509,1,47.33,6.5,
2,8,4,1,0,0,0,1,2,0,229,0\\r\\n15572390,1,27.83,4,1,3,8,5.75,1,1,2,1,2,75,1,0\\r\\n15801441,0,35
.75,0.915,2,6,4,0.75,1,1,4,0,2,0,1584,1\\r\\n15783859,1,33.58,0.25,2,3,5,4,0,0,0,1,1,420,1,0\\r\\
n15575243,1,34.08,0.08,1,7,5,0.04,1,1,1,1,2,280,2001,1\\r\\n15773421,0,20.75,10.335,2,13,8,0
.335,1,1,1,1,2,80,51,1\\r\\n15788776,1,33.17,1,2,14,4,0.75,1,1,7,1,2,340,4072,1\\r\\n15765257,

1,22.75,11,2,11,4,2.5,1,1,7,1,2,100,810,1\\r\\n15661412,1,48.75,8.5,2,8,8,12.5,1,1,9,0,2,181,16
56,1\\r\\n15636478,1,40.58,5,2,8,4,5,1,1,7,0,2,0,3066,1\\r\\n15603683,1,20.67,0.835,1,8,4,2,0,0
,0,1,1,240,1,0\\r\\n15651868,0,38.75,1.5,2,1,1,0,0,0,0,0,2,76,1,0\\r\\n15815443,0,57.08,19.5,2,8
,4,5.5,1,1,7,0,2,0,3001,1\\r\\n15682686,0,31.25,3.75,2,13,8,0.625,1,1,9,1,2,181,1,1\\r\\n156974
60,1,22,0.79,2,9,4,0.29,0,1,1,0,2,420,284,0\\r\\n15748432,0,58.33,10,2,11,4,4,1,1,14,0,2,0,1603
,1\\r\\n15698271,1,28.92,0.375,2,8,4,0.29,0,0,0,0,2,220,141,0\\r\\n15808662,0,46,4,2,5,3,0,1,0,
0,0,2,100,961,1\\r\\n15690372,1,21,3,1,2,4,1.085,1,1,8,1,2,160,2,1\\r\\n15781875,0,24.75,12.5,
2,6,4,1.5,1,1,12,1,2,120,568,1\\r\\n15801473,0,20.83,0.5,1,10,2,1,0,0,0,0,2,260,1,0\\r\\n157045
09,1,24.58,13.5,1,1,1,0,0,0,0,0,2,184,1,0\\r\\n15694666,0,26.5,2.71,1,8,4,0.085,0,0,0,0,1,80,1,0
\\r\\n15731166,1,40.92,0.835,2,1,1,0,1,0,0,0,2,130,2,0\\r\\n15728523,0,38.33,4.415,2,8,4,0.125,
0,0,0,0,2,160,1,0\\r\\n15788442,1,19.58,0.585,2,1,1,0,0,1,3,0,2,350,770,0\\r\\n15689781,1,39.2
5,9.5,2,7,4,6.5,1,1,14,0,2,240,4608,1\\r\\n15764226,1,25.75,0.5,2,8,4,1.46,1,1,5,1,2,312,1,1\\r\\
n15809837,0,46.08,3,2,8,4,2.375,1,1,8,1,2,396,4160,1\\r\\n15805212,1,19.67,10,1,4,8,0.835,1,
0,0,1,2,140,1,0\\r\\n15716082,0,22.25,1.25,1,1,1,3.25,0,0,0,0,2,280,1,0\\r\\n15643056,1,18.83,3
.54,1,1,1,0,0,0,0,1,2,180,2,0\\r\\n15654859,1,64.08,20,2,14,8,17.5,1,1,9,1,2,0,1001,1\\r\\n15761
158,0,16.5,1.25,2,11,4,0.25,0,1,1,0,2,108,99,0\\r\\n15577515,0,68.67,15,2,10,9,0,1,1,14,0,2,0,3
377,1\\r\\n15723827,1,76.75,22.29,2,10,9,12.75,1,1,1,1,2,0,110,1\\r\\n15646594,0,15.92,2.875,2
,11,4,0.085,0,0,0,0,2,120,1,0\\r\\n15712877,1,34.83,4,2,2,5,12.5,1,0,0,1,2,184,1,0\\r\\n15598802
,0,47.42,8,2,10,5,6.5,1,1,6,0,2,375,51101,1\\r\\n15699340,1,23.17,0,2,8,4,0,0,0,0,0,3,184,1,1\\r\\
n15691150,1,45.17,1.5,2,8,4,2.5,1,0,0,1,2,140,1,0\\r\\n15608688,1,15.17,7,2,10,4,1,0,0,0