```
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Invistigation of the Google dataset."
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "The aim of this prototype to take a prophylactic approach and improve the revenue of undervalued Google's apps to motivate the developers to keep working on them.\n",
    "\n",
    "Since her department's budget for marketing won't allow her to invest on ads (which would boost the number of sales), the only way to improve the revenue is by tweaking the price. She requests that you determine which paid apps are undervalued (undervalued here means that their price could be increased without lowering demand).\n",
    "The prototype conducted in order to achieve:\n",
    "\n",
    "1.Having a prototype makes it much easier to estimate the cost of a fully fleshed-out project, be it human, technological, or financial resources.\n",
    "\n",
    "2.Having a more accurate estimate of the cost allows decision-makers to not go through with the project if it doesn't seem profitable enough, or not the best allocation of resources at that point in time.\n",
    "\n",
    "3.It allows grunt workers (like ourselves) to change and add missing requirements to complete the project. Oftentimes these are overlooked and only spotted later; some examples are missing data, the creation of a new database, the development of an API.\n",
    "\n",
    "4.It allows business people (like the account manager) to tweak the project's goal.\n",
    "\n",
    "5.It gives all stakeholders an opportunity to add easily accomplishable side-goals given the main goal, thus maximizing the project's output"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "(10841, 13)\n"
     ]
    }
   ],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "\n",
    "playstore = pd.read_csv(\"googleplaystore.csv\")\n",
    "print(playstore.shape)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 30,
   "metadata": {},
   "outputs": [],
   "source": [
    "playstore.drop(labels=10472, inplace=True)\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 31,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "<class 'pandas.core.frame.DataFrame'>\n",
      "Int64Index: 10840 entries, 0 to 10840\n",
      "Data columns (total 13 columns):\n",
      " #   Column          Non-Null Count  Dtype  \n",
      "---  ------          --------------  -----  \n",
      " 0   App             10840 non-null  object \n",
      " 1   Category        10840 non-null  object \n",
      " 2   Rating          9366 non-null   float64\n",
      " 3   Reviews         10840 non-null  object \n",
      " 4   Size            10840 non-null  object \n",
      " 5   Installs        10840 non-null  object \n",
      " 6   Type            10839 non-null  object \n",
      " 7   Price           10840 non-null  object \n",
      " 8   Content Rating  10840 non-null  object \n",
      " 9   Genres          10840 non-null  object \n",
      " 10  Last Updated    10840 non-null  object \n",
      " 11  Current Ver     10832 non-null  object \n",
      " 12  Android Ver     10838 non-null  object \n",
      "dtypes: float64(1), object(12)\n",
      "memory usage: 1.2+ MB\n"
     ]
    }
   ],
   "source": [
    "playstore.info()\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "We notice that there are several columns which should have a numeric format but whose type is object. Specifically, Reviews, Size, and Price. We've saved you the trouble of exploring the issues with these columns."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 32,
   "metadata": {},
   "outputs": [],
   "source": [
    "playstore['Price'] = playstore['Price'].str.replace(\"$\",\"\").astype(float)\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Now that Price has been dealt with, we can focus on the paid apps"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 33,
   "metadata": {},
   "outputs": [
    {
     "data": {
```

```
"text/html": [
 "<div>\n",
 "<style scoped>\n",
 "    .dataframe tbody tr th:only-of-type {\n",
 "        vertical-align: middle;\n",
 "    }\n",
 "\n",
 "    .dataframe tbody tr th {\n",
 "        vertical-align: top;\n",
 "    }\n",
 "\n",
 "    .dataframe thead th {\n",
 "        text-align: right;\n",
 "    }\n",
 "</style>\n",
 "<table border=\"1\" class=\"dataframe\">\n",
 "  <thead>\n",
 "    <tr style=\"text-align: right;\">\n",
 "      <th></th>\n",
 "      <th>App</th>\n",
 "      <th>Category</th>\n",
 "      <th>Rating</th>\n",
 "      <th>Reviews</th>\n",
 "      <th>Size</th>\n",
 "      <th>Installs</th>\n",
 "      <th>Type</th>\n",
 "      <th>Price</th>\n",
 "      <th>Content Rating</th>\n",
 "      <th>Genres</th>\n",
 "      <th>Last Updated</th>\n",
 "      <th>Current Ver</th>\n",
 "      <th>Android Ver</th>\n",
 "    </tr>\n",
 "  </thead>\n",
 "  <tbody>\n",
 "    <tr>\n",
 "      <th>234</th>\n",
 "      <td>TurboScan: scan documents and receipts in PDF</td>\n",
 "      <td>BUSINESS</td>\n",
 "      <td>4.7</td>\n",
 "      <td>11442</td>\n",
 "      <td>6.8M</td>\n",
 "      <td>100,000+</td>\n",
 "      <td>Paid</td>\n",
 "      <td>4.99</td>\n",
    "      <td>Everyone</td>\n",
    "      <td>Business</td>\n",
    "      <td>March 25, 2018</td>\n",
    "      <td>1.5.2</td>\n",
    "      <td>4.0 and up</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>235</th>\n",
    "      <td>Tiny Scanner Pro: PDF Doc Scan</td>\n",
    "      <td>BUSINESS</td>\n",
    "      <td>4.8</td>\n",
    "      <td>10295</td>\n",
    "      <td>39M</td>\n",
    "      <td>100,000+</td>\n",
    "      <td>Paid</td>\n",
    "      <td>4.99</td>\n",
    "      <td>Everyone</td>\n",
    "      <td>Business</td>\n",
    "      <td>April 11, 2017</td>\n",
    "      <td>3.4.6</td>\n",
    "      <td>3.0 and up</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>290</th>\n",
    "      <td>TurboScan: scan documents and receipts in PDF</td>\n",
    "      <td>BUSINESS</td>\n",
    "      <td>4.7</td>\n",
    "      <td>11442</td>\n",
    "      <td>6.8M</td>\n",
    "      <td>100,000+</td>\n",
    "      <td>Paid</td>\n",
    "      <td>4.99</td>\n",
    "      <td>Everyone</td>\n",
    "      <td>Business</td>\n",
    "      <td>March 25, 2018</td>\n",
    "      <td>1.5.2</td>\n",
    "      <td>4.0 and up</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>291</th>\n",
    "      <td>Tiny Scanner Pro: PDF Doc Scan</td>\n",
    "      <td>BUSINESS</td>\n",
    "      <td>4.8</td>\n",
    "      <td>10295</td>\n",
    "      <td>39M</td>\n",
       "      <td>100,000+</td>\n",
       "      <td>Paid</td>\n",
       "      <td>4.99</td>\n",
       "      <td>Everyone</td>\n",
       "      <td>Business</td>\n",
       "      <td>April 11, 2017</td>\n",
       "      <td>3.4.6</td>\n",
       "      <td>3.0 and up</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>427</th>\n",
       "      <td>Puffin Browser Pro</td>\n",
       "      <td>COMMUNICATION</td>\n",
       "      <td>4.0</td>\n",
       "      <td>18247</td>\n",
       "      <td>Varies with device</td>\n",
       "      <td>100,000+</td>\n",
       "      <td>Paid</td>\n",
       "      <td>3.99</td>\n",
       "      <td>Everyone</td>\n",
       "      <td>Communication</td>\n",
       "      <td>July 5, 2018</td>\n",
       "      <td>7.5.3.20547</td>\n",
       "      <td>4.1 and up</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "                                               App       Category  Rating  \\\n",
       "234  TurboScan: scan documents and receipts in PDF       BUSINESS     4.7   \n",
       "235                 Tiny Scanner Pro: PDF Doc Scan       BUSINESS     4.8   \n",
       "290  TurboScan: scan documents and receipts in PDF       BUSINESS     4.7   \n",
       "291                 Tiny Scanner Pro: PDF Doc Scan       BUSINESS     4.8   \n",
       "427                             Puffin Browser Pro  COMMUNICATION     4.0   \n",
       "\n",
       "    Reviews                Size  Installs  Type  Price Content Rating  \\\n",
       "234   11442                6.8M  100,000+  Paid   4.99       Everyone   \n",
       "235   10295                 39M  100,000+  Paid   4.99       Everyone   \n",
       "290   11442                6.8M  100,000+  Paid   4.99       Everyone   \n",
       "291   10295                 39M  100,000+  Paid   4.99       Everyone   \n",
       "427   18247  Varies with device  100,000+  Paid   3.99       Everyone   \n",
       "\n",
       "            Genres    Last Updated  Current Ver Android Ver  \n",
       "234       Business  March 25, 2018        1.5.2  4.0 and up  \n",
       "235       Business  April 11, 2017        3.4.6  3.0 and up  \n",
       "290       Business  March 25, 2018        1.5.2  4.0 and up  \n",
       "291       Business  April 11, 2017        3.4.6  3.0 and up  \n",
       "427  Communication    July 5, 2018  7.5.3.20547  4.1 and up  "
      ]
     },
     "execution_count": 33,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "paid = playstore.loc[playstore['Price'] !=0].copy()\n",
    "\n",
    "#Alternatively we can run\n",
    "#paid = playstore[playstore[\"Type\"] == \"Paid\"].copy()\n",
    "paid.head(5)\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Drop the Type column from paid"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 34,
   "metadata": {},
   "outputs": [],
   "source": [
    "\n",
    "paid.drop('Type', axis=\"columns\", inplace=True)\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 35,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "<class 'pandas.core.frame.DataFrame'>\n",
      "Int64Index: 800 entries, 234 to 10798\n",
      "Data columns (total 12 columns):\n",
      " #   Column          Non-Null Count  Dtype  \n",
      "---  ------          --------------  -----  \n",
      " 0   App             800 non-null    object \n",
      " 1   Category        800 non-null    object \n",
      " 2   Rating          647 non-null    float64\n",
      " 3   Reviews         800 non-null    int64  \n",
      " 4   Size            724 non-null    float64\n",
      " 5   Installs        800 non-null    object \n",
      " 6   Price           800 non-null    float64\n",
      " 7   Content Rating  800 non-null    object \n",
      " 8   Genres          800 non-null    object \n",
      " 9   Last Updated    800 non-null    object \n",
      " 10  Current Ver     798 non-null    object \n",
      " 11  Android Ver     799 non-null    object \n",
      "dtypes: float64(3), int64(1), object(8)\n",
      "memory usage: 81.2+ KB\n"
     ]
    }
   ],
   "source": [
    "def clean_size(size):\n",
    "#      \"\"\"Convert file size string to float and megabytes\"\"\"\n",
    "    size = size.replace(\"M\",\"\")\n",
    "    if size.endswith(\"k\"):\n",
    "        size = float(size[:-1])/1000\n",
    "    if size =='Varies with device':\n",
    "        size = np.NaN\n",
    "    \n",
    "    else:\n",
    "        size = float(size)\n",
    "    return size\n",
    "\n",
    "\n",
    "paid[\"Reviews\"] = paid[\"Reviews\"].astype(int)\n",
    "paid[\"Size\"] = paid[\"Size\"].apply(clean_size).astype(float)\n",
    "paid.info()\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Dropping the duplicate values."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 36,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "The number of rows before dropping duplicates 800:\n"
     ]
    }
   ],
   "source": [
    "\n",
    "print('The number of rows before dropping duplicates {}:'.format(paid.shape[0]))\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 37,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>App</th>\n",
"      <th>Category</th>\n",
"      <th>Rating</th>\n",
"      <th>Reviews</th>\n",
"      <th>Size</th>\n",
"      <th>Installs</th>\n",
"      <th>Price</th>\n",
"      <th>Content Rating</th>\n",
"      <th>Genres</th>\n",
"      <th>Last Updated</th>\n",
"      <th>Current Ver</th>\n",
"      <th>Android Ver</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>10735</th>\n",
"      <td>FP VoiceBot</td>\n",
"      <td>FAMILY</td>\n",
"      <td>NaN</td>\n",
"      <td>17</td>\n",
"      <td>0.157</td>\n",
"      <td>100+</td>\n",
"      <td>0.99</td>\n",
"      <td>Mature 17+</td>\n",
"      <td>Entertainment</td>\n",
"      <td>November 25, 2015</td>\n",
"      <td>1.2</td>\n",
"      <td>2.1 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>10760</th>\n",
"      <td>Fast Tract Diet</td>\n",
"      <td>HEALTH_AND_FITNESS</td>\n",
"      <td>4.4</td>\n",
"      <td>35</td>\n",
"      <td>2.400</td>\n",
"      <td>1,000+</td>\n",
"      <td>7.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Health &amp; Fitness</td>\n",
"      <td>August 8, 2018</td>\n",
"        <td>1.9.3</td>\n",
"        <td>4.2 and up</td>\n",
"      </tr>\n",
"      <tr>\n",
"        <th>10782</th>\n",
"        <td>Trine 2: Complete Story</td>\n",
"        <td>GAME</td>\n",
"        <td>3.8</td>\n",
"        <td>252</td>\n",
"        <td>11.000</td>\n",
"        <td>10,000+</td>\n",
"        <td>16.99</td>\n",
"        <td>Teen</td>\n",
"        <td>Action</td>\n",
"        <td>February 27, 2015</td>\n",
"        <td>2.22</td>\n",
"        <td>5.0 and up</td>\n",
"      </tr>\n",
"      <tr>\n",
"        <th>10785</th>\n",
"        <td>sugar, sugar</td>\n",
"        <td>FAMILY</td>\n",
"        <td>4.2</td>\n",
"        <td>1405</td>\n",
"        <td>9.500</td>\n",
"        <td>10,000+</td>\n",
"        <td>1.20</td>\n",
"        <td>Everyone</td>\n",
"        <td>Puzzle</td>\n",
"        <td>June 5, 2018</td>\n",
"        <td>2.7</td>\n",
"        <td>2.3 and up</td>\n",
"      </tr>\n",
"      <tr>\n",
"        <th>10798</th>\n",
"        <td>Word Search Tab 1 FR</td>\n",
"        <td>FAMILY</td>\n",
"        <td>NaN</td>\n",
"        <td>0</td>\n",
"        <td>1.020</td>\n",
"        <td>50+</td>\n",
"        <td>1.04</td>\n",
"        <td>Everyone</td>\n",
"        <td>Puzzle</td>\n",
"        <td>February 6, 2012</td>\n",
      "      <td>1.1</td>\n",
      "      <td>3.0 and up</td>\n",
      "    </tr>\n",
      "  </tbody>\n",
      "</table>\n",
      "</div>"
     ],
     "text/plain": [
      "                           App            Category  Rating  Reviews    Size  \\\n",
      "10735              FP VoiceBot              FAMILY     NaN       17   0.157   \n",
      "10760          Fast Tract Diet  HEALTH_AND_FITNESS     4.4       35   2.400   \n",
      "10782  Trine 2: Complete Story                GAME     3.8      252  11.000   \n",
      "10785             sugar, sugar              FAMILY     4.2     1405   9.500   \n",
      "10798     Word Search Tab 1 FR              FAMILY     NaN        0   1.020   \n",
      "\n",
      "      Installs  Price Content Rating            Genres       Last Updated  \\\n",
      "10735     100+   0.99     Mature 17+     Entertainment  November 25, 2015   \n",
      "10760   1,000+   7.99       Everyone  Health & Fitness     August 8, 2018   \n",
      "10782  10,000+  16.99           Teen            Action  February 27, 2015   \n",
      "10785  10,000+   1.20       Everyone            Puzzle       June 5, 2018   \n",
      "10798      50+   1.04       Everyone            Puzzle   February 6, 2012   \n",
      "\n",
      "      Current Ver Android Ver  \n",
      "10735         1.2  2.1 and up  \n",
      "10760       1.9.3  4.2 and up  \n",
      "10782        2.22  5.0 and up  \n",
      "10785         2.7  2.3 and up  \n",
      "10798         1.1  3.0 and up  "
     ]
    },
    "execution_count": 37,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "paid.tail(5)\n"
  ]
 },
 {
  "cell_type": "raw",
  "metadata": {},
  "source": [
   "We'll save you the trouble of inspecting these rows. Each duplicate exists for one of two reasons:\n",
   "\n",
    "    Everything is equal except the category\n",
    "    Everything is equal except the number of reviews\n",
    "\n",
    "The instances of the first reason are the apps Fuzzy Numbers: Pre-K Number Foundation and Toca Life: City, which both appear with the categories EDUCATION and FAMILY. Investigating the apps on the Play Store leads us to conclude that the FAMILY category isn't correct, so we'll get rid of these rows:\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 38,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th>App</th>\n",
       "      <th>Category</th>\n",
       "      <th>Rating</th>\n",
       "      <th>Reviews</th>\n",
       "      <th>Size</th>\n",
       "      <th>Installs</th>\n",
       "      <th>Price</th>\n",
       "      <th>Content Rating</th>\n",
       "      <th>Genres</th>\n",
       "      <th>Last Updated</th>\n",
       "      <th>Current Ver</th>\n",
     "      <th>Android Ver</th>\n",
     "    </tr>\n",
     "  </thead>\n",
     "  <tbody>\n",
     "    <tr>\n",
     "      <th>2151</th>\n",
     "      <td>Toca Life: City</td>\n",
     "      <td>FAMILY</td>\n",
     "      <td>4.7</td>\n",
     "      <td>31100</td>\n",
     "      <td>24.0</td>\n",
     "      <td>500,000+</td>\n",
     "      <td>3.99</td>\n",
     "      <td>Everyone</td>\n",
     "      <td>Education;Pretend Play</td>\n",
     "      <td>July 6, 2018</td>\n",
     "      <td>1.5-play</td>\n",
     "      <td>4.4 and up</td>\n",
     "    </tr>\n",
     "    <tr>\n",
     "      <th>4301</th>\n",
     "      <td>Fuzzy Numbers: Pre-K Number Foundation</td>\n",
     "      <td>FAMILY</td>\n",
     "      <td>4.7</td>\n",
     "      <td>21</td>\n",
     "      <td>44.0</td>\n",
     "      <td>1,000+</td>\n",
     "      <td>5.99</td>\n",
     "      <td>Everyone</td>\n",
     "      <td>Education;Education</td>\n",
     "      <td>July 21, 2017</td>\n",
     "      <td>1.3</td>\n",
     "      <td>4.1 and up</td>\n",
     "    </tr>\n",
     "  </tbody>\n",
     "</table>\n",
     "</div>"
    ],
    "text/plain": [
     "                                         App Category  Rating  Reviews  Size  \\\n",
     "2151                         Toca Life: City   FAMILY     4.7    31100  24.0   \n",
     "4301  Fuzzy Numbers: Pre-K Number Foundation   FAMILY     4.7       21  44.0   \n",
     "\n",
     "      Installs  Price Content Rating                  Genres   Last Updated  \\\n",
     "2151  500,000+   3.99       Everyone  Education;Pretend Play   July 6, 2018   \n",
       "4301    1,000+   5.99       Everyone     Education;Education  July 21, 2017   \n",
       "\n",
       "     Current Ver Android Ver  \n",
       "2151    1.5-play  4.4 and up  \n",
       "4301         1.3  4.1 and up  "
      ]
     },
     "execution_count": 38,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "app_mask =paid['App'].isin(['Fuzzy Numbers: Pre-K Number Foundation', \"Toca Life: City\"])\n",
    "category_mask = paid[\"Category\"]=='FAMILY'\n",
    "\n",
    "paid[app_mask& category_mask]\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Now that we've identified the labels, we can drop these rows:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 39,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "The number of rows before dropping duplicates 44:\n"
     ]
    }
   ],
   "source": [
    "print('The number of rows before dropping duplicates {}:'.format(paid[\"App\"].duplicated().sum()))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 40,
   "metadata": {},
   "outputs": [],
   "source": [
    "paid.drop([2151,4301],inplace=True)\n",
    "\n",
    "# paid.drop(labels=[2151,4301],inplace=True)\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 41,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "The number of rows after dropping duplicates 42:\n"
     ]
    }
   ],
   "source": [
    "print('The number of rows after dropping duplicates {}:'.format(paid.duplicated(subset=[\"App\"]).sum()))\n"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Reset the index of paid.\n",
    "    Pass True to the drop argument in order to not save the old index."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 42,
   "metadata": {},
   "outputs": [],
   "source": [
    "paid.reset_index(inplace=True, drop=True)\n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Identify the research request."
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Note that we still haven't defined what \"undervalued apps\" means here. The science in data science is partially justified by
the fact that some kinds of problems we encounter need research to even be properly defined. We need to get to know the data before
contemplating what proxy to use to approximate the concept of undervalued.\n",
    "\n",
    "Since this request is centered on the price, we'll start with that column. Let's create a histogram for this column:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 43,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f96e4c8e2d0>]],\n",
       "      dtype=object)"
      ]
     },
     "execution_count": 43,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAsUAAAF1CAYAAAAA6ZfwAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAAemklEQVR4nO3df6xnZ30f+PcnOJCESTw2P0aObcW0jNKktuLaN8S7SNUdnG1sp429ElYdWWAjZ
6cruZQ06cKkWintqlnIrrZukVqkaZwyVEkmLgm1F5s07MBsSiWTMOBgwI0YwIGxXTsEYzJhkxT62T/uucnt5Y7nnvG993vN83pJV99znvOc73m+H5175z3nPvec6u4AAMDI
vmXRAwAAgEUTigEAGJ5QDADA8IRiAACGJxQDADA8oRgAgOEJxQDPI1X1vqq6bdHjAPhmU+5TDLBYVfVokn1Jvp7kj5M8kOSN3X16keMCGIkrxQC7w9/q7j1Jrkrygn0U
ba4Wf2QDbxA9YgF2kux9L8r4kl1fV8ar6uar6j0m+muQvTW0/sdq/qv6nqnqkqv6oqj5VVVdN7d9dVb9WVX9QVZ+rqr+3mE8E8PwgFAPsIlV1aZIbknxsanpdkoNJvjPJ76
/re3OSf5Tk9Um+K8mPJfnD6Yry/53kd5NcnOTaJD9ZVT+yAx8B4HlJKAbYHf5dVX05yYeS/L9J/vep/Z3d/cnu/lp3/5d1+/xEkv+ju3+nV5zs7t/PyvSLl3X3/9bdf9bdn
03yr5LcslMfBuD55rxFDwCAJMlN3f3/rG2oqiT5wrPsc2mSz2zQ/j1JvnsK2atekOQ/PNdBAnyzEooBdrdnu0XQF5L85TO0f66792/PkAC++Zg+AfD89QtJ/kFVXT3dneKV
VfU9SX47yVeq6i1V9e1V9YKquryqfnDB4wXYtYRigOep7v63SX4uyS8n+aMk/y7Jhd399SR/K8mVST6X5ItZCdDnL2ioALueh3cAADA8V4oBABieUAwAwPCEYgAAhicUAwA
wPKEYAIDh7YqHd7z0pS/tyy67bEeP+cd//Md58YtfvKPHfD5Tr/nUbB71mke95lGv+dRsHvWaZ5H1OnHixBe7+2Xr8zviqvq7yf5iaw8WenhJG9IclGSo0kuTPLRJK/r7j
```

+rqhcleVeSq5P8YZK/3d2PPtv7X3bZZfnIRz6y+U+zBY4fP57l5eUdPebzmXrNp2bzqNc86jWPes2nZvOo1zyLrFdV/f5G7WedPlFVFyf5e0mWuvvyJC9IckuSn09y1/QY0
aeT3DHtckeSp7v7lUnumvoBAMCutdk5xecl+faqOi/JdyR5Islrkrx72n4kyU3T8o3Teqbt11ZVbc1wAQBg623qiXZV9aasPEr0/0vym0nelOTB6WpwqurSJO/r7sur6hNJ
ruvuU9O2zyT5oe7+4rr3PJjkYJLs27fv6qNHj27dp9qE06dPZ8+ePTt6zOcz9ZpPzeZRr3nUax71mk/N5lGveRZZrwMHDpzo7qX17WedU1xVF2Tl6u8rknw5yb9Ncv0GXVf
T9UZXhb8heXf34SSHk2Rpaal3el6JuT/zqNd8ajaPes2jXXvOo13xqNo96zbMb67WZ6RM/nORz3f0H3f1fkvx6kv8+yd5pOkWSXJLk8Wn5VJLk2Tafn6SL23pqAEAYAttJh
R/Psk1VfUd09zga5N8KskHk7x26nNbknun5fum9UzbP9CbmaMBAAALctZQ3N0fzsofzH00K7dj+5asTHt4S5KfqqqTSV6S5O5pl7uTvGRq/6kkh7Zh3AAAsGU2dZ/i7v7ZJ
D+7rvmzSV61Qd8/SXLzcx8aAADsDI95BgBgeEIxAADDE4oBABieUAwAwPCEYgAAhrepu098M7vs0P2LHsKOevRtP7roIQAA7DquFAMAMDyhGACA4QnFAAAMTygGAGB4QjEA
AMMTigEAGJ5QDADA8IRiAACGJxQDADA8oRgAgOEJxQAADE8oBgBgeEIxAADDE4oBABieUAwAwPCEYgAAhicUAwAwPKEYAIDhCcUAAAxPKAYAYHhCMQAAwxOKAQAYnlAMAMD
whGIAAIYnFAMAMLyzhuKq+t6qemjN11eq6ier6sKqen9VfXp6vWDqX1X19qo6WVUfr6qrtv9jAADAuTtrKO7u3+vuK7v7yiRXJ/lqkvckOZTkWHfvT3JsWk+S65Psn74OJn
nHdgwcAAC2ytzpE9cm+Ux3/36SG5McmdqPJLlpWr4xybt6xYNJ9lbVRVsyWgAA2AZzQ/EtSX5lWt7X3U8kyfT68qn94iRfWLPPqakNAAB2peruzXWsemGSx5P81e5+sqq+3
N1712x/ursvqKr7k7y1uz80tR9L8ubuPrHu/Q5mZXpF9u3bd/XRo0e35hNt0unTp7Nnz548/NgzO3rcRbvi4vPPab/VerF5ajaPes2jXvOo13xqNo96zbPIeh04cOBEdy+t
bz9vxntcn+Sj3f3ktP5kVV3U3U9M0yOemtpPJbl0zX6XZCVM/ze6+3CSw0mytLTUy8vLM4by3B0/fjzLy8u5/dD9O3rcRXv01uVz2m+1Xmyems2jXvOo1zzqNZ+azaNe8+z
Ges2ZPvHj+YupE0lyX5LbpuXbkty7pv31010orknyzOo0CwAA2I02daW4qr4jyf+Q5O+saX5bknuq6o4kn09y89T+QJIbkpzMyp0q3rBlowUAgG2wqVDc3V9N8pJ1bX+Ylb
tRrO/bSe7cktEBAMAO8EQ7AACGJxQDADA8oRgAgOEJxQAADE8oBgBgeEIxAADDE4oBABieUAwAwPCEYgAAhicUAwAwPKEYAIDhCcUAAAxPKAYAYHhCMQAAwxOKAQAYnlAMA
MDwhGIAAIYnFAMAMDyhGACA4QnFAAAMTygGAGB4QjEAAMMTigEAGJ5QDADA8IRiAACGJxQDADA8oRgAgOEJxQAADE8oBgBgeEIxAADDE4oBABjepkJxVe2tqndX1X+qqkeq
6r+rqgur6v1V9enp9YKpb1XV26vqZFV9vKqu2t6PAAAAz81mrxT/8yS/0d1/JckPJHHkkyaEkx7p7f5Jj03qSXJ9k//R1MMk7tnTEAACwxc4aiqvqu5L89SR3J0l3/1l3fzn
JjUmOTN2OJLlpWr4xybt6xYNJ9lbVRVs+cgAA2CKbuVL8l5L8QZJ/XVUfq6pfqKoXJ9nX3U8kyfT68qn/xUm+sGb/U1MbAADsStXdz96hainJg0le3d0frqp/nuQrSd7Y3X
vX9Hu6uy+oqvuTvLW7PzS1H0vy5u4+se59D2ZlekX27dt39dGjR7fyc53V6dOns2fPnjz82DM7etxFu+Li889pv9V6sXlqNo96zaNe86jXfGo2j3rNs8h6HThw4ER3L61vP
28T+55Kcqq7Pzytvzsr84efrKqLuvuJaXrEU2v6X7pm/0uSPL7+Tbv7cJLDSbK0tNTLy8ub/Sxb4vjx41leXs7th+7f0eMu2qO3Lp/TfqvM4h9KtRRcPDX1IbjsGlIQIB5pH3rWHEUVEaXCJ51bsPPaH7ehYvwtaIDtqx2Pb7B6M4WWb0NLxkxpRjk5YHmubtZCMOVaPdkSQHi+Kw+dTiiNX6c0cT6qXyk5Mpns/1Ttv0F++/6fT3lJIP9Vqx+MDKYbRbZ8ETl1Qi/KIO+i8tH

```
9MiptenpvZTSS5ds/slSR7fmuECAMDW28zdJyrJ3Uke6e5/umbTfUlum5ZvS3LvmvbXT3ehuCbJM939xBaOGQAAttRmpk+8OsnrkjxcVQ9Nbf8wyduS3FNVdyT5fJKbp20P
JLkhyckkX03yhi0dMQAAbLGzhuLpD+bONIH42g36d5I7n+O4AABgx3iiHQAAwxOKAQAYnlAMAMDwhGIAAIYnFAMAMDyhGACA4QnFAAAMTygGAGB4QjEAAMMTigEAGJ5QDAD
A8IRiAACGJxQDADA8oRgAgOEJxQAADE8oBgBgeEIxAADDE4oBABieUAwAwPCEYgAAhicUAwAwPKEYAIDhCcUAAAxPKAYAYHhCMQAAwxOKAQAYnlAMAMDwhGIAAIYnFAMAMD
yhGACA4QnFAAAM76yhuKp+saqeqqpPrGm7sKreX1Wfnl4vmNqrqt5eVSer6uNVddV2Dh4AALbCZq4UvzPJdevaDiU51t37kxyb1pPk+iT7p6+DSd6xNcMEAIDtc9ZQ3N2/l
eRL65pvTHJkWj6S5KY17e/qFQ8m2VtVF23VYAEAYDuc65zifd39RJJMry+f2i9O8oU1/U5NbQAAsGtVd5+9U9VlSd7b3ZdP61/u7r1rtj/d3RdU1f1J3trdH5rajyV5c3ef
2OA9D2ZlikX27dt39dGjR7fg42ze6dOns2fPnjz82DM7etxFu+Li889pv9V6sXlqNo96zaNe86jXfGo2j3rNs8h6HThw4ER3L61vP+8c3+/Jqrqou5+Ypkc8NbWfSnLpmn6
XJHl8ozfo7sNJDifJ0tJSLy8vn+NQzs3x48ezvLyc2w/dv6PHXbRHb10+p/1W68Xmqdk86jWPes2jXvOp2TzqNc9urNe5Tp+4L8lt0/JtSe5d0/766S4U1yR5ZnWaBQAA7F
ZnvVJcVb+SZDnJS6vqVJKfTfK2JPdU1R1JPp/k5qn7A0luSHIyyVeTvGEbxgwAAFvqrKG4u3/8DJuu3aBvJ7nzuQ4KAAB2kifaAQAwPKEYAIDhCcUAAAxPKAYAYHhCMQAAw
xOKAQAYnlAMAMDwhGIAAIYnFAMAMDyhGACA4QnFAAAMTygGAGB4QjEAAMMTigEAGJ5QDADA8IRiAACGJxQDADA8oRgAgOEJxQAADE8oBgBgeEIxAADDE4oBABieUAwAwPCE
YgAAhicUAwAwPKEYAIDhCcUAAAxPKAYAYHjnLXoAAAAju+zQ/Ysewo5753UvXvQQvoErxQAADE8oBgBgeNsSiqvquqr6vao6WVWHtuMYAACwVbY8FFfVC5L8iyTXJ/n+JD9
eVd+/1ccBAICtsh1/aPeqJCe7+7NJUlVHk9yY5FPbcCxmOtfJ/D99xddy+/P0DwEefduPLnoIAMAutx2h+OIkX1izfirJD23DcWBTFvVXvc/n/0gsgnrN89NXfC3Lix4EO2
K0n2EuZLAo1d1b+4ZVNyf5ke7+iWn9dUle1d1vXNfvYJKD0+r3Jvm9LR3I2b00yRd3+JjPZ+o1n5rNo17zqNc86jWfms2jXvMssl7f090vW9+4HVeKTyW5dM36JUkeX9+pu
w8nObwNx9+UqvpIdy8t6vjPN+o1n5rNo17zqNc86jWfms2jXvPsxnptx90nfifJ/qp6RVW9MMktSe7bhuMAAMCW2PIrxd39tar6u0n+fZIXJPnF7v7kVh8HAAC2yrY85rm7
H0jywHa89xZa2NSN5yn1mk/N5lGvedRrHvWaT83mUa95dl29tvwP7QAA4PnGY54BABjekKHYY6jPrqoeraqHq+qhqvrI1HZhVb2/qj49vV6w6HEuSlX9YlU9VVWfWNO2YX1
qxdun8+3jVXXV4ka+GGeo1z+qqsemc+yhqrphzbafmer1e1X1I4sZ9eJU1aVV9cGqeqSqPllVb5ranWNn8Cw1c55toKq+rap+u6p+d6rXP57aX1FVH57OsV+d/mA+VfWiaf
3ktP2yRY5/pz1Lvd5ZVZ9bc35dObUP/z2ZrDzluKo+VlXvndZ39fk1XCguj6Ge40B3X7nmlimHkhzr7v1Jjk3ro3pnkuvWtZ2pPtcn2T99HUzyjh0a427yznxjvZLkrukcu
3L6W4RM34+3JPmr0z7/cvq+HcnXkvx0d39fkmuS3DnVxTl2ZmeqWeI828ifJnlNd/9AkiuTXFdV1yT5+azUa3+Sp5PcMfW/I8nT3f3KJHdN/UZypnolyf+y5vx6aGrzPbni
TUkeWbO+q8+v4UJx1jyGurv/LMnqY6g5uxuTHJmWjyS5aYFjWaju/q0kX1rXfKb63JjkXb3iwSR7q+qinRnp7nCGep3JjUmOdvefdvfnkpzMyvftMLr7ie7+6LT8R1n5R+X
iOMfO6FlqdiZDn2fTuXJ6Wv3W6auTvCbJu6f29efY6rn37iTXVlXt0HAX7lnqdSbDf09W1SVJfjTJL0zrlV1+fo0Yijd6DPWz/eAcVSf5zao6UStPH0ySfd39RLLyD1CSly
9sdLvTmerjnDuzvzv9avEX6y+m46jXGtOvEf9akg/HObYp62qWOM82NP1q+6EkTyV5f5LPJPlyd39t6rK2Jn9er2n7M0lesrMjXqz19eru1fPr56bz666qetHUNvz5leSfJ
Xlzkv86rb8ku/z8GjEUb/Q/D7fg+Eav7u6rsvIroDur6q8vekDPY865jb0jyV/Oyq8in0jyf03t6jWpqj1Jfi3JT3b3V56t6wZtarZSM+fZGXT317v7yqw8efZVSb5vo27T
q3qtq1dVXZ7kZ5L8lSQ/mOTCJG+Zug9dr6r6m0me6u4Ta5s36Lqrzq8RQ/GmHkM9uu5+fHp9Ksl7svID88nVX/9Mr08tboS70pnq45zbQHc/Of0j81+T/Kv8xa+u1StJVX1
rVsLdL3X3r0/NzrFnsVHNnGdn191fTnI8K3Ox91bV6jMM1tbkz+s1bT8/m58S9U1lTb2um6btdHf/aZJ/HefXqlcn+bGqejQr01Rfk5Urx7v6/BoxFHsM9VlU1Yur6jtXl5
P8jSSfyEqdbpu63Zbk3sWMcNc6U33uS/L66a+Rr0nyzOqvwEe2bn7d/5iVcyxZqdct018jvyIrf6jy2zs9vkWa5tLdneSR7v6nazY5x87gTDVznm2sql5WVXun5W9P8sNZm
Yf9wSSvnbqtP8dWz73XJvlAD/SggzPU6z+t+U9qZWV+7Nrza9jvye7+me6+pLsvy0rO+kB335pdfn5tyxPtdjOPod6UfUneM81xPy/JL3f3b1TV7yS5p6ruSPL5JDcvcIwL
VVW/kmQ5yUur6lSSn03ytmxcnweS3JCVP+T5apI37PiAF+wM9Vqebl/USR5N8neSpLs/WVX3JPlUVu4ocGd3f30R416gVyd5XZKHpzmMSfIP4xx7Nmeq2Y87zzZ0UZIj0x0
3viXJPd393qr6VJKjVfVPknwsK//RyPT6b6rqZFau4N2yiEEv0Jnq9YGqellWfv3/UJL/eerve3Jjb8kuPr880Q4AgOGNOH0CAAD+G0IxAADDE4oBABieUAwAwPCEYgAAhi
cUAwAwPKEYAIDhCcUAAAzv/werEtbpgtYM5gAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 864x432 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "paid.hist('Price', grid=True, figsize=(12,6))\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "It seem that The distributions of prices is heavily skewed to the right and we have a few outliers."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 44,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th>App</th>\n",
       "      <th>Category</th>\n",
       "      <th>Rating</th>\n",
       "      <th>Reviews</th>\n",
       "      <th>Size</th>\n",
       "      <th>Installs</th>\n",
       "      <th>Price</th>\n",
       "      <th>Content Rating</th>\n",
       "      <th>Genres</th>\n",
       "      <th>Last Updated</th>\n",
       "      <th>Current Ver</th>\n",
       "      <th>Android Ver</th>\n",
       "    </tr>\n",
       "  </thead>\n",
       "  <tbody>\n",
"    <tr>\n",
"      <th>213</th>\n",
"      <td>I'm Rich - Trump Edition</td>\n",
"      <td>LIFESTYLE</td>\n",
"      <td>3.6</td>\n",
"      <td>275</td>\n",
"      <td>7.300</td>\n",
"      <td>10,000+</td>\n",
"      <td>400.00</td>\n",
"      <td>Everyone</td>\n",
"      <td>Lifestyle</td>\n",
"      <td>May 3, 2018</td>\n",
"      <td>1.0.1</td>\n",
"      <td>4.1 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>200</th>\n",
"      <td>most expensive app (H)</td>\n",
"      <td>FAMILY</td>\n",
"      <td>4.3</td>\n",
"      <td>6</td>\n",
"      <td>1.500</td>\n",
"      <td>100+</td>\n",
"      <td>399.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Entertainment</td>\n",
"      <td>July 16, 2018</td>\n",
"      <td>1.0</td>\n",
"      <td>7.0 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>349</th>\n",
"      <td>I am rich (Most expensive app)</td>\n",
"      <td>FINANCE</td>\n",
"      <td>4.1</td>\n",
"      <td>129</td>\n",
"      <td>2.700</td>\n",
"      <td>1,000+</td>\n",
"      <td>399.99</td>\n",
"      <td>Teen</td>\n",
"      <td>Finance</td>\n",
"      <td>December 6, 2017</td>\n",
"      <td>2</td>\n",
"      <td>4.0.3 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>212</th>\n",
"      <td>💎 I'm rich</td>\n",
"      <td>LIFESTYLE</td>\n",
"      <td>3.8</td>\n",
"      <td>718</td>\n",
"      <td>26.000</td>\n",
"      <td>10,000+</td>\n",
"      <td>399.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Lifestyle</td>\n",
"      <td>March 11, 2018</td>\n",
"      <td>1.0.0</td>\n",
"      <td>4.4 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>354</th>\n",
"      <td>I AM RICH PRO PLUS</td>\n",
"      <td>FINANCE</td>\n",
"      <td>4.0</td>\n",
"      <td>36</td>\n",
"      <td>41.000</td>\n",
"      <td>1,000+</td>\n",
"      <td>399.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Finance</td>\n",
"      <td>June 25, 2018</td>\n",
"      <td>1.0.2</td>\n",
"      <td>4.1 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>...</th>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>168</th>\n",
"      <td>B-52 Spirits of Glory Deluxe</td>\n",
"      <td>GAME</td>\n",
"      <td>4.3</td>\n",
"      <td>12</td>\n",
"      <td>29.000</td>\n",
"      <td>100+</td>\n",
"      <td>0.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Arcade</td>\n",
"      <td>September 2, 2017</td>\n",
"      <td>1.5.9</td>\n",
"      <td>2.3 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>288</th>\n",
"      <td>X Back - Icon Pack</td>\n",
"      <td>PERSONALIZATION</td>\n",
"      <td>4.5</td>\n",
"      <td>56</td>\n",
"      <td>26.000</td>\n",
"      <td>10,000+</td>\n",
"      <td>0.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Personalization</td>\n",
"      <td>June 29, 2018</td>\n",
"      <td>1.6.2</td>\n",
"      <td>4.1 and up</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>563</th>\n",
"      <td>CT and XR Dose Calculator</td>\n",
"      <td>MEDICAL</td>\n",
"      <td>NaN</td>\n",
"      <td>3</td>\n",
"      <td>0.097</td>\n",
"      <td>50+</td>\n",
"      <td>0.99</td>\n",
"      <td>Everyone</td>\n",
"      <td>Medical</td>\n",
"      <td>January 22, 2014</td>\n",
"      <td>2014.01</td>\n",
"      <td>1.6 and up</td>\n",
"    </tr>\n",
       "    <tr>\n",
       "      <th>559</th>\n",
       "      <td>Emergency Brain CT</td>\n",
       "      <td>MEDICAL</td>\n",
       "      <td>NaN</td>\n",
       "      <td>2</td>\n",
       "      <td>19.000</td>\n",
       "      <td>10+</td>\n",
       "      <td>0.99</td>\n",
       "      <td>Everyone</td>\n",
       "      <td>Medical</td>\n",
       "      <td>April 17, 2018</td>\n",
       "      <td>1.0.0</td>\n",
       "      <td>4.1 and up</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>252</th>\n",
       "      <td>Q Alerts: QAnon Drop Notifications, Research +++</td>\n",
       "      <td>NEWS_AND_MAGAZINES</td>\n",
       "      <td>4.7</td>\n",
       "      <td>143</td>\n",
       "      <td>26.000</td>\n",
       "      <td>5,000+</td>\n",
       "      <td>0.99</td>\n",
       "      <td>Mature 17+</td>\n",
       "      <td>News &amp; Magazines</td>\n",
       "      <td>July 26, 2018</td>\n",
       "      <td>4.1.10</td>\n",
       "      <td>4.1 and up</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "<p>798 rows × 12 columns</p>\n",
       "</div>"
      ],
      "text/plain": [
       "                                                  App            Category  \\\n",
       "213                          I'm Rich - Trump Edition           LIFESTYLE   \n",
       "200                            most expensive app (H)              FAMILY   \n",
       "349                    I am rich (Most expensive app)             FINANCE   \n",
       "212                                        💎 I'm rich           LIFESTYLE   \n",
       "354                                I AM RICH PRO PLUS             FINANCE   \n",
       "..                                                ...                 ...   \n",
       "168                      B-52 Spirits of Glory Deluxe                GAME   \n",
       "288                                X Back - Icon Pack     PERSONALIZATION   \n",
         "563                         CT and XR Dose Calculator             MEDICAL   \n",
         "559                                Emergency Brain CT             MEDICAL   \n",
         "252  Q Alerts: QAnon Drop Notifications, Research +++  NEWS_AND_MAGAZINES   \n",
         "\n",
         "     Rating  Reviews    Size Installs   Price Content Rating  \\\n",
         "213     3.6      275   7.300  10,000+  400.00       Everyone   \n",
         "200     4.3        6   1.500     100+  399.99       Everyone   \n",
         "349     4.1      129   2.700   1,000+  399.99           Teen   \n",
         "212     3.8      718  26.000  10,000+  399.99       Everyone   \n",
         "354     4.0       36  41.000   1,000+  399.99       Everyone   \n",
         "..      ...      ...     ...      ...     ...            ...   \n",
         "168     4.3       12  29.000     100+    0.99       Everyone   \n",
         "288     4.5       56  26.000  10,000+    0.99       Everyone   \n",
         "563     NaN        3   0.097      50+    0.99       Everyone   \n",
         "559     NaN        2  19.000      10+    0.99       Everyone   \n",
         "252     4.7      143  26.000   5,000+    0.99     Mature 17+   \n",
         "\n",
         "               Genres       Last Updated Current Ver   Android Ver  \n",
         "213         Lifestyle        May 3, 2018       1.0.1    4.1 and up  \n",
         "200     Entertainment      July 16, 2018         1.0    7.0 and up  \n",
         "349           Finance   December 6, 2017           2  4.0.3 and up  \n",
         "212         Lifestyle     March 11, 2018       1.0.0    4.4 and up  \n",
         "354           Finance      June 25, 2018       1.0.2    4.1 and up  \n",
         "..                ...                ...         ...           ...  \n",
         "168            Arcade  September 2, 2017       1.5.9    2.3 and up  \n",
         "288   Personalization      June 29, 2018       1.6.2    4.1 and up  \n",
         "563           Medical   January 22, 2014     2014.01    1.6 and up  \n",
         "559           Medical     April 17, 2018       1.0.0    4.1 and up  \n",
         "252  News & Magazines      July 26, 2018      4.1.10    4.1 and up  \n",
         "\n",
         "[798 rows x 12 columns]"
        ]
       },
       "execution_count": 44,
       "metadata": {},
       "output_type": "execute_result"
      }
     ],
     "source": [
      "paid.sort_values('Price',ascending=False)"
     ]
    },
    {
     "cell_type": "markdown",
     "metadata": {},
   "source": [
    "It seems these apps are not only very niche, but they also would spoil our data, given how disparate the prices are from the others.\n",
    "\n",
    "Looking back at the histogram, it's clear that the vast majority of apps have a price significantly below $50. Let's restrict our analysis to these rows:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 45,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f96e51d6290>]],\n",
       "      dtype=object)"
      ]
     },
     "execution_count": 45,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAeYAAAE/CAYAAACTomAoAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAASzUlEQVR4nO3dfayedX3H8fdnLajxqTwcCLbF4mw2jBkPq6wLy+Jg2XhQ2z9sgnHSkLr+wxxGj
VazxGnmAvtDkGQhYeAsi08MH+gU3ViBqUtAiyCK1VAR6VkZPcqDIFEHfvfH/Ws4aQ+cm9Nzc369z/uVnNzX9bt+5z6/c0X79rru+9ymqpAkSX34rYVegCRJepphliSpI4ZZ
kqSOGGZJkjpimCVJ6ohhliSpI4ZZWoSSfCXJxoVeh6QDxb9jlsZDkvuAY4GngF8ANwDvqKrHF3Jdkp4br5il8fLGqnoJcCrwOuBvph/MgP+9lzrmf0GlMVRV/wN8BXhtklu
SfCTJfwNPAK9qY2/fNz/JXybZmeSxJN9Pcmobf0WSzyWZSvLjJH+9ML+RtHgYZmkMJVkJnAPc0YbeBmwGXgr8ZL+5G4C/Bc4HXga8CfhZu7L+N+A7wHLgTOCdSf78efgVpE
XLMEvj5YtJHgG+AfwX8Pdt/BNVdXdVPVlV/7ff97wd+Ieq+lYN7KqqnzC4FT5RVR+uql9X1b3APwHnPV+/jLQYLV3oBUiaV+ur6j+nDyQB2P0s37MS+NEM468EXtFCv88S4
OsHu0hJz8wwS4vDs/35xW7gt59h/MdVtXo0S5I0E29lS7oKeE+S32/v2n51klcC3wR+nuR9SV6UZEmS1yZ53QKvVxprhlla5KrqX4GPAJ8CHgO+CBxZVU8BbwROBn4M/JRB
xF++QEuVFgU/YESSpI54xSxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUke6+ICRo48+ulatWrXQy5Ak6Xlx++23/7SqJmY61kWYV61axY4dOxZ6GZIkPS+S/OSZjnkrW5Kkjhh
mSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSepIF5+VPd9WbfnyQi9hVvddfO5CL0GS1CGvmCVJ6ohhliSpI4ZZkqSODBXmJMuSXJ
fkB0l2JvnDJEcmuTHJPe3xiDY3SS5PsivJXUlOHe2vIEnS+Bj2ivljwFer6neBk4CdwBZge1WtBra3fYCzgdXtazNwxbyuWJKkMTZrmJO8DPhj4GqAqvp1VT0CrAO2tmlbg
fVtex1wTQ3cCixLcty8r1ySpDE0zBXzq4Ap4J+T3JHkqiQvBo6tqgcA2uMxbf5yYPe0759sY5IkaRbDhHkpcCpwRVWdAvyCp29bzyQzjNUBk5LNSXYk2TE1NTXUYiVJGnfD
hHkSmKyq29r+dQxC/eC+W9Ttce+0+Sunff8KYM/+T1pVV1bVmqpaMzExMdf1S5I0VmYNc1X9L7A7ye+0oTOB7wPbgI1tbCNwfdveBpzf3p29Fnh03y1vSZL07Ib9SM53AJ9
McjhwL3ABg6hfm2QTcD+woc29ATgH2AU80eZKkqQhDBXmqroTWDPDoTNnmFvAhQe5LkmSFiU/+UuSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhl
mSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqi
GGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4MFeYk9yX5bpI7k+xoY0cmuTHJPe3xiDaeJJcn2ZXkriSnjvIXkCRpnDyXK+Y/qaqTq2pN298CbK+q1cD2tg9wNrC6
fW0GrpivxUqSNO4O5lb2OmBr294KrJ82fk0N3AosS3LcQfwcSZIWjWHDXMB/JLk9yeY2dmxVPQDQHo9p48uB3dO+d7KNSZKkWSwdct7pVbUnyTHAjUl+8CxzM8NYHTBpEPj
NAMcff/yQy5AkabwNdcVcVXva417gC8BpwIP7blG3x71t+iSwctq3rwD2zPCcV1bVmqpaMzExMfffQJKkMTJrmJO8OMlL920DfwZ8D9gGbGzTNgLXt+1twPnt3dlrgUf33f
KWJEnPbphb2ccCX0iyb/6nquqrSb4FXJtkE3A/sKHNvwE4B9gFPAFcMO+rliRpTM0a5qq6FzhphvGfGAWfOMF7AhfOyOkmSFhk/+UuSpI4YZkmSOmKYJUnqiGGWJKkjhlmSp
I4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGW
JKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4YZkmSOmKYJUnqiGGWJKkjhlmSpI4MHeYkS5LckeRLbf+EJLcluSfJZ5Mc3sZf0PZ3teOrRrN0SZLGz3O5Yr4I2Dlt/xL
g0qpaDTwMbGrjm4CHq+rVwKVtniRJGsJQYU6yAjgXuKrtBzgDuK5N2Qqsb9vr2j7t+JltviRJmsWwV8yXAe8FftP2jwIeqaon2/4ksLxtLwd2A7Tjj7b5kiRpFrOGOckbgL
1Vdfv04Rmm1hDHpj/v5iQ7kuyYmpoaarGSJI27Ya6YTwfelOQ+4DMMbmFfBixLsrTNWQHsaduTwEqAdvzlwEP7P2lVXVlVa6pqzcTExEH9EpIkjYtZw1xV76+qFVW1CjgPu
Kmq3grcDLy5TdsIXN+2t7V92vGbquqAK2ZJknSgg/k75vcB70qyi8FryFe38auBo9r4u4AtB7dESZIWj6WzT3laVd0C3NK27wVOm2HOL4EN87A2SZIWHT/5S5KkjhhmSZI6
YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZK
kjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjswa5iQvTPLNJN9JcneSD7XxE5LcluSeJJ9Ncngbf0Hb39
WOrxrtryBJ0vgY5or5V8AZVXUScDJwVpK1wCXApVW1GngY2NTmbwIerqpXA5e2eZIkaQizhrkGHm+7h7WvAs4ArmvjW4H1bXtd26cdPzNJ5m3FkiSNsaFeY06yJMmdwF7gR
uBHwCNV9WSbMgksb9vLgd0A7fijwFHzuWhJksbVUGGuqqeq6mRgBXAacOJM09rjTFfHtf9Aks1JdiTZMTU1Nex6JUkaa8/pXdlV9QhwC7AWWJZkaTu0AtjTtieBlQDt+MuB
h2Z4riurak1VrZmYmJjb6iVJGjPDvCt7Ismytv0i4E+BncDNwJvbtI3A9W17W9unHb+pqg64YpYkSQdaOvsUjgO2JlnCIOTXVtWXknwf+EySvwPuAK5u868G/iXJLgZXyue
NYN2SJI2lWcNcVXcBp8wwfi+D15v3H/8lsGFeVidJ0iLjJ39JktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJHTHMkiR1xD
BLktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJHTHMkiR1xDBLktQRwyxJUkcMsyRJH
THMkiR1ZNYwJ1mZ5OYkO5PcneSiNn5kkhuT3NMej2jjSXJ5kl1J7kpy6qh/CUmSxsUwV8xPAu+uqhOBtcCFSV4DbAG2V9VqYHvbBzgbWN2+NgNXzPuqJUkaU7OGuaoeqKpv
t+3HgJ3AcmAdsLVN2wqsb9vrgGtq4FZgWZLj5n3lkiSNoef0GnOSVcApwG3AsVX1AAziDRzTpi0Hdk/7tsk2JkmSZjF0mJO8BPgc8M6q+vmzTZ1hrGZ4vs1JdiTZMTU1New
yJEkaa0OFOclhDKL8yar6fBt+cN8t6va4t41PAiunffsKYM/+z1lVV1bVmqpaMzExMdf1S5I0VoZ5V3aAq4GdVfXRaYe2ARvb9kbg+mnj57d3Z68FHt13y1uSJD27pUPMOR
14G/DdJHe2sQ8AFwPXJtkE3A9saMduAM4BdgFPABfM64olSRpjs4a5qr7BzK8bA5w5w/wCLjzIdUmStCj5yV+SJHXEMEuS1BHDLElSRwyzJEkdMcySJHXEMEuS1BHDLElSR
wyzJEkdMcySJHXEMEuS1BHDLElSRwyzJEkdMcySJHXEMEuS1BHDLElSRwyzJEkdMcySJHXEMEuS1BHDLElSRwyzJEkdMcySJHXEMEuS1BHDLElSRwyzJEkdMcySJHXEMEuS
1BHDLElSRwyzJEkdMcySJHXEMEuS1BHDLElSRwyzJEkdMcySJHVk1jAn+XiSvUm+N23syCQ3JrmnPR7RxpPk8iS7ktyV5NRRLl6SpHEzzBXzJ4Cz9hvbAmyvqtXA9rYPcDa
wun1tBq6Yn2VKkrQ4zBrmqvoa8NB+w+uArW17K7B+2vg1NXArsCzJcfO1WEmSxt1cX2M+tqoeAGiPx7Tx5cDuafMm25gkSRrCfL/5KzOM1YwTk81JdiTZMTU1Nc/LkCTp0D
TXMD+47xZ1e9zbxieBldPmrQD2zPQEVXVlVa2pqjUTExNzXIYkSeNlrmHeBmxs2xuB66eNn9/enb0WeHTfLW9JkjS7pbNNSPJp4PXA0UkmgQ8CFwPXJtkE3A9saNNvAM4Bd
gFPABeMYM2SJI2tWcNcVW95hkNnzjC3gAsPdlGSJC1Ws4ZZo7Fqy5cXegmzuu/icxd6CZK06PiRnJIkdcQwS5LUEcMsSVJHDLMkSR0xzJIkdcQwS5LUEcMsSVJHDLMkSR0x
zJIkdcQwS5LUEcMsSVJHDLMkSR0xzJIkdcQwS5LUEcMsSVJHDLMkSR0xzJIkdcQwS5LUEcMsSVJHDLMkSR0xzJIkdcQwS5LUEcMsSVJHDLMkSR0xzJIkdcQwS5LUEcMsSVJ
HDLMkSR1ZutALkOZq1ZYvL/QSZnXfxecu9BIkHWK8YpYkqSNeMesZHQpXpJI0brxiliSpIyO5Yk5yFvAxYAlwVVVdPIqfI/XOuw7jz/cRaL7N+xVzkiXAPwJnA68B3pLkNf
P9cyRJGkejuJV9GrCrqu6tql8DnwHWjeDnSJI0dkZxK3s5sHva/iTwByP4OZK04Hy5YnF4Pl+yGEWYM8NYHTAp2QxsbruPJ/nhc/gZRwM/ncPaNBzP72h5fkfL8zt6i+4c5
5J5f8pXPtOBUYR5Elg5bX8FsGf/SVV1JXDlXH5Akh1VtWZuy9NsPL+j5fkdLc/v6HmOR2sUrzF/C1id5IQkhwPnAdtG8HMkSRo7837FXFVPJvkr4N8Z/LnUx6vq7vn+OZIk
jaOR/B1zVd0A3DCK527mdAtcQ/P8jpbnd7Q8v6PnOR6hVB3wvixJkrRA/EhOSZI6ckiFOclZSX6YZFeSLQu9nnGQ5ONJ9ib53rSxI5PcmOSe9njEQq7xUJZkZZKbk+xMcne
Si9q453geJHlhkm8m+U47vx9q4yckua2d38+2N6JqjpIsSXJHki+1fc/vCB0yYfajPkfmE8BZ+41tAbZX1Wpge9vX3DwJvLuqTgTWAhe2/9x6jufHr4Azquok4GTgrCRrgU
uAS9v5fRjYtIBrHAcXATun7Xt+R+iQCTN+1OdIVNXXgIf2G14HbG3bW4H1z+uixkhVPVBV327bjzH4x205nuN5UQOPt93D2lcBZwDXtXHP70FIsgI4F7iq7QfP70gdSmGe6
aM+ly/QWsbdsVX1AAzCAhyzwOsZC0lWAacAt+E5njftNuudwF7gRuBHwCNV9WSb4r8VB+cy4L3Ab9r+UXh+R+pQCvNQH/Up9SjJS4DPAe+sqp8v9HrGSVU9VVUnM/iUwdOA
E2ea9vyuajwkeQOwt6punz48w1TP7zwayd8xj8hQH/WpefFgkuOq6oEkxzG4EtEcJTmMQZQ/WVWfb8Oe43lWVY8kuYXBa/nLkixtV3X+WzF3pwNvSnIO8ELgZQyuoD2/I3Q
oXTH7UZ/Pn23Axra9Ebh+AddySGuvx10N7Kyqj0475DmeB0kmkixr2y8C/pTB6/g3A29u0zy/c1RV76+qFVW1isG/uTdV1Vvx/I7UIfUBI+1/tV3G0x/1+ZEFXtIhL8mngd
cz+H+LeRD4IPBF4FrgeOB+YENV7f8GMQ0hyR8BXwe+y9Ov0X2AwevMnuODlOT3GLz5aAmDC41rq+rDSV7F4A2iRwJ3AH9RVb9auJUe+pK8HnhPVb3B8ztah1SYJUkad4fSr
WxJksaeYZYkqSOGWZKkjhhmSZI6YpglSeqIYZYkqSOGWZKkjhhmSZI68v8mAIgUt8/EFQAAAABJRU5ErkJggg==\n",
       "text/plain": [
        "<Figure size 576x360 with 1 Axes>"
       ]
      },
      "metadata": {
       "needs_background": "light"
      },
      "output_type": "display_data"
     }
    ],
    "source": [
     "\n",
    "affordable_apps = paid[paid[\"Price\"]<50].copy()\n",
    "affordable_apps.hist(column=\"Price\", grid=False, figsize=(8,5))\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 48,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0       True\n",
       "1       True\n",
       "2       True\n",
       "3       True\n",
       "4       True\n",
       "       ...  \n",
       "793     True\n",
       "794    False\n",
       "795    False\n",
       "796     True\n",
       "797     True\n",
       "Name: Price, Length: 774, dtype: bool"
      ]
     },
     "execution_count": 48,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Create a mask called cheap to identify the apps in affordable_apps that cost less than $5.\n",
    "cheap = affordable_apps[\"Price\"] < 5\n",
    "\n",
    "# Create a mask called reasonable to identify the apps in affordable_apps that cost $5 or more.\n",
    "reasonable = affordable_apps[\"Price\"] >= 5\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 49,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f96e56d1750>]],\n",
       "      dtype=object)"
      ]
     },
     "execution_count": 49,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAsYAAAF1CAYAAADr3izzAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAAXz0lEQVR4nO3de7DuV13f8fe3OeJdAXOgmKQe1GiLjBc8plimDgVbQZQwU53GaSVabEaLt1arQ
TvSOrWDtqPW2tqJQgmtcileiAJWRBTtFPAEvADRkkKEI2iOInjroNHVP/YTsz3s5JyzL2fvJK/XzJ7n+a3fevbzPWvW7Odz1l7795u1VgAAcH/3Vw67AAAAOAoEYwAASDAG
AIBKMAYAgEowBgCASjAGAIBKMAa4V5mZl8/MtYddB8B90biOMcDhmpnbqodWf1b9Uf/Wy6qvWWn94mHUB3N9YMQY4Gj5/rfVh1aOqz6j+5faTs8XPbIAD5IcswBGy1vrN6uX
VI2fmZ2fm22bmf1V/XH3spu3L7uw/M/9kZm6ZmT+YmTfPzKM27R89Mz88M2dm5m0z89WH8y8CuPcQjAGOkJm5ovrc6g2bpi+urqs+vPqNs/p+YfWvqqdWH1E9ufrdzcryj
e/XF1WPb762pn5nIvwTwC41xKMAY6GH5uZ91S/UP1c9W837c9da71prXXHWutz7rPl1Xfsdb6xbXl1rXWb7S1FeP4Wutb11p/stZ6a/X91TUX6x8DcG907LALALAKCqp6y1f
np7w8xUveMeXnNF9X93aP+Y6qM3QftOl1Q/v9ciAe7LBGOAo+2eLh30jurj7qb9bWutKw+mJD7JlspAO69fqD6+pn59M1VKz5+Zj6mel31+zPzjTPzwTNzycw8cmY+45Dr
BTjSBGOAe6m11v+ovq36oeoPqh+rHrzW+rPq86tPrd5W/U5bIfojD6lUgHsFN/gAAICsGAMAQCUYAwBAJRgDAEAlGAMAQCUYAwBAdURu8HHppZeuEydOHHYZZAADcx918882
/s9Y6vtO5IxGMT5w40alTpw67DAAA7uNm5jfu5w5t1LMzHNm5vaZeeNZZ7V81M78+M2+ame/Y1v6v72rAue/uS11vtm5iGb9k
dU11SfVH109dMz8wmbuzABAMCRdc4V47XWq6t3n9X8X8FdWz1lrv2/S5+dWdWz1lrv2/S5fdN+dfWCtdb71lpvq26trtrHegEA4EDs9qoUn1D97x8Mx8xqb9suod2/qd3rS9n5m5bmZO
cypM2fO7LIMAADYH7sNxseqB1WPrv5F9/PgqGmPC2H9/xDwMBAOCi2W0wPl39yNryuurPq0s37Vds63d59c69lQGAAAdvt8H4x6rHVc3MJ1QPqH6n
uqm6ZmY+cGYeXl1ZvW4/4fYn1bVrrVW9aWZeVL25uqN6fuwdVuaW8aL25uqN6uitSAABwbzBbefZwnTx5crnBBwAAB21mbl5rndzp3G63UUgAAwH2
KYAwAAAnGAABQCcYAAFCdx1Up7utOXP/Swy7t8vz6tWn33WygPjjr6rBIAAI4cK8YAAJBgDAAAlWAMAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAA
CVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAA1XkE45l5zszcPjNv3OHc18/MmplLN8czMy+Oc18/MmplLN8czM98zM
7fOzK/MzKMOomgAANhv57Ni/NzqCWc3zswV1RdqQCWc3zswV1d+t3r6t+YnVlZuv66rv23uJAABw8M4ZjNdar67evcOp/7bx/Oz27uY51vcPaflAi7+aY8+Rqdp5CgAAfnuUMx0+zA9U/KQMZhvxNvHPw76vWCA4zEMqxgg
qcuqd2w7Pr1p2+l7XDczp2bm1JkzzZ3ZTBgAA7JsLDsYz8yHVN1ffstPpHdrWTm1rrdtqwrqj/qaAyBYAAD76tguXvNx1cOrX56Zqsur18/MVW2tEF+xre/l1Tv2WiQAABy0C14xXmv96lrroWutE2utE22F4UettX6ruqt6lzPKEAg/97GX9+
AAByWC14xXmv96lrrIWutE2utE22F4UettX6ruql66ubqFI+u3rvWetf+lgwAAPvvfC7X9OCCBsZPpZAnwQ6Nrd2wcBvkbrmQwSUvWSWl90jvMntj1f1d
P3XhYAAFxc7nwHAAAJxgAAUAnGAABQCcYAAFAJxgAAUAnGAABQCcYAAFAJxgAAUJ3HDT4AOJpOXP/Swy7horvtWU7867BKA+zArxgAAkGAMAACVYAwAAJVgDAAAlWAMAACVY
AwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAA1XkE45l5zszcPjNv3Nb2
72bm12bmV2bmR2fmgdvOPWNmbp2ZX5+ZzzzmowgEAYD+dz4rxc6snnNX2iuqRa61Prv5P9YzqmXlEdU31SZvX/OeZuWTfqgUAgANyzmC81np19e6z2n5qrXXH5vA1bY0R7+VXH5vA11eWb51d
XL1hrvW+t9bbq1uaaawXAAAOxH7sMSf7H1cs3ve3rf7ezTH1o+rP711IGAADs2bHdRddKN2AGAADs2bHdvn
Bmrq0+r3r8WuvwO8Hu6umJbt8urd+6+PAAAuDh2tWI8M0+ovrF3z+ovrF68lrrj7eduqm6zZmY+cGYeXl1ZvW7vZQIAwME654rxrxzdy/emx16cycrp7Z1lUoPrB6xcxUvat9eVrTfNz
IuqN7e1xeLpa60/O6jiAQ8Bgv5wzGK+1vmiH5H9/9vq75tL0UBAMDF5s53AACQYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVg
DAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAAD
VeQTjmXnOzNw+M2/c1vbgmXnFzLxl8/ygbeeeMTO3zs3Tz+TzhaPi9Zbc5aS+HT/W3yH9oMS+4QEzpCa5IO9iG/zw0Z2o5MkA2ZLh5+h9ZV5u2v+b2vOWv3K3WF1cBHdm3HIr+HHcAj+0d0zZVdGKR9x7c+mcgKbnAuNUHTmgk7L3
dWT9nW/ry15TXVA2fmYftVLAAAHJTd7jF+6FrrXVWt9a7rJvWcyBuSt/PztSqE87/ngqT1dsF5jyZnP8xt29x5ir5KtmkqKnfg6rs+yxx7DwG3pSt6cBE6HSMyM9u/wFF+2nFwMg1Oy+L5CutPqJlgj7LBwZmJtU7AXwrbZhPIp7jMcMxhn9OTF/xz
9wisXm8fdN+urpiW7/Lq3fuvjwAALg4dhuMb6qu3ty+Xrrb8VmOeJWg
d85gvNb6ors59fgd+q6qnr7XogAA4GLb7eXaAADgPkUwBgCABGMAAKgEYwAAqARjAACoBGMAAKgEYwAAqARjAACoBGMAAKgEYwAAqARjAACoB
EYwAAqARjAACoBGMAAKgEYwAAqARjAACoBGMAAKgEYwAAqARjAACoBGMAAKgEYwAAqARjAACoBGMAAKgEYwAAqPYYjGfmn5f1rhCvaHSpkTJNZcnV
l5+My8dmbeMjMvnJkH7FfFwAAOBB2fUeQVx6D/1YeGMgwZAbOgdSPe7/9V2G9f48gG7MbwGHdLwc9Bcm0pjY9FMzc7bBMW0ZEKJRqYF2Mf5+SQVD43AWOlJxOqbB65KE31sm1xBkfcRXmVMP4Vlqd9ndPnJq9HJhhb0qfu3
lYgfm91c/WetdYdm26nqbGwA5pWe3DPsHvgKfGrTCW63K1IDsgM9PkAA8F0e6lEJoE/cSgrT/AqlktI6gzzqBCgwo9VMPtCvgMa+5b1ZftUnSeVDx0WMjOfmPNSw7tbD/WtGrTLMNHkxT5IZjaoWuBGeUoTuMg1KMeXthf9n5eVIb+0GfDfG+fqxT1jRNWBYAFb2q3Be
s7Wi6vLqnXusEQAADtiWr6Ee8XP8+PPg7sF4Tc0mTUgsdGYCPvLA4U6Cn5Fwj+MQkS89zbxzdW7DMV8xn9AzZLXeeJJxpvB9oNtxFn1xLfnE79DbD0RY+XQ4C9OYYf1QtA
```

```
J59TRsetgXLXWemb1zLOa31pdtZfvCwAAF5s73wEAQIIxAABUgjEAAFSCMQAAVIIxAABUgjEAAFSCMQAAVIIxAABUgjEAAFSCMQAAVIIxAABUgjEAAFSCMQAAVIIxAABUgj
EAAFSCMQAAVIIxAABUgjEAAFSCMQAAVIIxAABUgjEAAFSCMQAAVIIxAABUdeywCwAA7t6J61962CVcdLc960mHXQL3U1aMAQAgwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAK
sEYAACqPQbjmXngzLx4Zn5tZm6Zmc+cmQfPzCtm5i2bxwftV7EAAHBQ9rpi/B+qn1xr/fXqU6pbquurV661rqxeuTkGAIAjbdfBeGY+ovqs6tlVa60/WWu9p7q6unHT7cbq
KXstEgAADtpeVow/tjpT/deZecPM/MDMfGj10LXWu6o2jw/ZhzoBAOBA7SUYH6seVX3fWuvTqj/qArZNzMx1M3NqZk6dOXNmD2UAAMDe7SUYn65Or7Veuzl+cVtB+bdn5mF
Vm8fbd3rxWuuGtdbJtdbJ48eP76EMAADYu10H47XWb1XvmJlP3DQ9vnpzdVN17abt2uole6oQAAAugmN7fP1XVT84Mw+o3lp9aVth+0Uz87Tq7dUX7vE9AADgwO0pGK+1fq
k6ucOpx+/l+wIAwMXmzncAAJBgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAAlWAMAADV3m8Jzb3QietfetglAAAcOVaMAQAgwRgAACrBGAAAKsEYAAAqwRgAA
CrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAqn0IxjNzycy8YWZ+YnP88Jl5
7cy8ZWZeODMP2HuZAABwsPZjxfhrqlu2HX979V1rrSur36uetg/vAQAAB2pPwXhmLq+eVP3A5niqx1Uv3nS5sXrKXt4DAAAuhr2uGH939Q3Vn2+OP6p6z1rrjs3x6eqyPb4
HAAAcuF0H45n5vOr2tdbN25t36Lru5vXXzcypmTl15syZ3ZYBAAD7Yi8rxo+pnjwzt1UvaGsLxXdXD5yZY5s+l1fv3OnFa60b1lon1lonjx8/vocyAABg73YdjNdaz1hrXb
7WOlFdU/3MWusfVq+qvmDT7drqJXuuEgAADthBXMf4G6t/PjO3trXn+NkH8B4AALCvjp27y7mttX62+tnN87dWV+3H9wUAgIvFne8AACDBGAAAKsEYAAAqwRgAACrBGAAAK
sEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEYAAAqwRgAAKo6dtgFAPvvxPUvPewSLrrbnvWkwy4BgHs5K8YA
AJBgDAAAlWAMAACVYAwAAJVgDAAAlWAMAACVYAwAAJVgDAAA1R6C8cxcMTOvmplbZuZNM/M1m/YHz8wrZuYtm8cH7V+5AABwMPayYnxH9XXrb9
RPbp6+sw8orq+euVa68rqlZtjAAA40nYdjNda71prvX7z/A+qW6rLqqurGzfdbqyestciAQDgoO3LHuOZOVF9XXa6qFrrXfWVniuHrIf7wEAAAdpz8F4Zj6s+uHqa9dav3
8Br7tuZk7NzKkzZ87stQwAANiTPQXjmfmAtkLxD661fmTT/Nsz87DN+YdVt+/02rXWDWutk2utk8ePH99LGQAAsGd7uSrFVM+ubllrfee2Uzd12UzdV126eX1u9ZPflAQDAxXFsD
699TPXF1a/OzC9t2r6pelb1opl5WvX26gv3ViIAABy8XQfjtdYvVHM3px+/2+8LAACHwZ3vAAAgwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAKsEY
AAAqwRgAACrBGAAAKsEYAAAqwRgAACrBGAAAqjp22AUAAGx34vqXHnYJ3E9ZMQYAgARjAACoBGMAAKjsMQbgXsTeU+AgWTEGAIAEYwAAqARjAAC
oBGMAAKhcxxi4j3B9WwD2yooxAAAkGAMAQCUYAwBAdYDBeGaeMDO/PjO3zsz1B/U+AACwHw4kGM/MJdV/qp5YPaL6rp1rfXWtdafVC+orj6g9w
IAgD07qGB8WfWObcenN20AAHAkHdR1jGeHtWxvXCV123OfzDmn1A6rlfFxa/c4hvv+9jfG6MMbrwhivC2fMLozxujDG68IYr/Mw3/4XTw9jvD7m7k4cVDA+XV2x7fjy6
p3bO6y1bqhuOKD3D3vyPr47sWztLFaKX6yunJmHkHz8wDqmuqmw7ovQYQAAYM8OZMV4rXXHzHxV9ZPVJdVz1lpvOoj3AgCA/XBQWylaa72setlBfX0AANhPB3aDDwAAOD+QRFQ5OTpG
++GgtlK01npZ9bKD+v777Ehs6bgXMV4XxnhdGON14YZhTFeF8Z4XRjjdWGO1HjNWuvcvQAA4D7OLaEBAKD7UTCemefMzO0z88a7T8z8z2bW1j/ysw86mLXeJScx3g9dmb
eOzO/tPn6lotd41EyM1fMzKtm5paZedZedPMfM0OfcyxjfMcL3NsY2Y+aGZeNzO/vBmvf71Dnw+cmRdu5tdrZ+bFzZ+YkdzplfZz
nHeJlfZ5mZ22bmVzfjcWqH80fiM/LA9hgfQc+tvrd63t2cf2J15ebrb1bft3m8v3pu9zxeVT+/1vq8i1P0OkXdXH9XXrrdfPzIdXN8/MK97Zab97Wxxy7y/mMV5ljd3pf9bi11
h/OzAdUvzAzL19rvWZbn6dVv7fW+viZuab69uofHEaxR8D5jFfFVC9daX3kI9R1VX1PdUn3EDufMr/d3T+NV5td0/s5a6+6uWXwkPiPvNyvGa61XV+++hy5VXV89bW15TPXBm
HnZxqjt6zmO82Gat9a611us3z/+grR+/Z9/Yy/+/gB6+z/0Dk6urGzfMXV4+fmZ1utnSfd57jxTYYzc3n1pOoH7qaL+bXNeYwXF+5IfEbeb4LxeXAb6wv
3mZtfVb58Zj7psIs5Kja/Yvy06rVnOn+TLHdnAP41Xm2F/Y/Nr2l6rbq1este52fq217qjeW33Uxa3y6DiP8ar6+5tf2b54Zq7Y4fz9yXdX31D9+d2cN7/+snONV5lfZ1vVT8
3MzbN19+OzHYnPSMH4Lue8jTV/yeurj1lrfUr1H6sfO+R6jBXv9/tmnd3LbnMV4/Xp1Ya31y9dPdtRp6v
zMzn1fdvta6+Z667dB2v5xf5zle5tf7e8xa61FtbZl4+sx81lnnj8QcE4zvcs7bWHOXtdbv3/mrys01qz9gZi495LIO1WYv4P7jW+pEjlCzN5VFeKK+L6Cb
s079xfyamWPVR2Y71N2011rrd9da79scfn/16Re5tKPM5tZWTZ+a26gXV42bmvwdx12bmv5/Vx/y6znHy/x6f2utd24eb69+tLrqrC5/rqC5H4jNSML7LTdVTN38V+ejqvWutdx12UZf+VzPz
VO/eXzcxVbc2l3z3cqg7PZY3eZmeMz88DN8w+uPrv6tbO2rvS45Lb2ud8vrbWesda6fK11orqmrbnzj87qZn
5tnM94mV9/2cx86OYPrZuZD63+XnX2Va+OxGfk/eaqFDPzQ9Vj2q3a/rf64b0Ozu6Oj1k+4LDqfRoI/x+oLqK2bmjur/VdcxX39Ibjym+uLqVzf7Gqu+q
fprZY7t4HzGyxy7y8OqG2fmkrb+g/Citdd ZPzMy3VqfWWje19R+N/zYzt7a1UnzN4ZV7pN3WdFYb1V+x0nf5k7vhjF56ZM58M5nvL56Zp7c1hy/u3rqoRV9ND26+uLqDZutwn
5zsAAMhWCgAAqARjAACoBGMAAKjsMQbgXsTeU+AgWTEGAIAEYwAAqARjAAC
```

```
      "text/plain": [
       "<Figure size 864x432 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    },
    {
     "data": {
      "image/png":
```

```
"iVBORw0KGgoAAAANSUhEUgAAAr8AAAF1CAYAAADhgoKhAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
```

B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAATzUlEQVR4nO3db6xkd33f8c+3XixoCLWNr62N7bLQrigoKobcWK6oqtQOrYMJ9gNcGaXpqnK6T
9IU1ETJJqrUJmqqpQ9C+qCqtMWUfUAAx/yxi0kaZ2M3SVUZ1tg0mA0yGAOuXe+FYGGCBLXz7YN73KzMLnf2zsydu/69XtLVzPnNGc135aP128fnzqnuDgAAjOCvrHoAAADY
KeIXAIBhiF8AAIYhfgEAGIb4BQBgGOIXAIBhiF+AXaiqfqeqDqx6DoAXmvI9vwA7o6oeTXJpkmeT/HmSTyTyT5ue7+1irnAhiJM78AO+snu/ulSd6Q5EeT/KtTX6xN/m4GWBJ
/wQKsQHf/7yS/k+SHq+reqvr1qvofSb6d5FXT2s88t39V/b0OqOlFVT1fV56rqDdP6D1XVh6tqo6q+VFX/YjV/IoBzg/gFWIGquiLJm5M8MC39MC39dJKDSX4wyZeft
+S5GVJ3prk69MZ4v+a5DNJLktybZJ3VtU/3IE/AsA5SfwC7KyPVdVTSf44yX9P8u+m9fd190Pd/Ux3/9/nvednkvz77v5Ub/pCd385m5dNrHX3r3X3d7v7kST/OcnNO/WHA
TjX7Fn1AACDubG7f//UhapKkq9+n/dckeSLp1l/RZIfmmL6Oecl+aN5hwR4oRK/ALvD9/vqna8m+RtnWP9Sd+9fzkgALzwuewDY/d6T5Beq6kemb4j4m1X1iiT3Nqvql
qnpJVZ1XVT9cVT+64nkBdi3xC7DLdfdvJ/n1JL+V5OkkkH0tyUXc/m+Qnk1yZ5EtJvpbNUP5rKxoVYNdzkwsAAIbhzC8AAMMQvwAADEP8AgAwDPELAMAwxC8AAMPY0ZtcXHz
xxb1v376d/EgAAAZz//33f62710732o7G7759+3L8+PGd/EgAAAZTVV8+02suewAAYBjiFwCAYYhfAACGIX4BABiG+AUAYBjiFwCAYY
hfAACGIX4BABiG+AUAYBh7Vj3ATth36K5Vj7DjHj18/apHAADYdZz5BQBgGOIXAIBhiF8AAIYhfgEAGMaW8VtVr66qB0/5+WUvbOqLqqqu6vq4enxwp0YGAAAtmvL+O3uz
3f3ld19ZZIfSfLtJB9NcijJse7en+TYtA0AALvW2V72cG2SL3SfPckOTotH40yY2LHAwAABBtbOP35iQfmjMn0eMkiBwMAgEWbOX6r6r6vwkb03y22fzAVV1sKqOO
V9XxjY2Ns50PAAAW5mzO/P/P5Ekk9395Pt9PT9395PT9pNVtTdJpseTp3tTdx/p7vXuXl9bW5tvWgAAmMPZxO/b85eXPCTJnUkOTM8M8PJLljUUMBAMAyzBS/VfVXk7wpyUdOWT6c5E1V9fD
02uHFjwcAAIuzZ5aduvvvbSV7+vLWvZ/PbHwAA4JzgDm8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAA
xD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvAADDEL8AAAxD/AIAMAzxCwDAMMQvA
ADDEL8AAAxD/AIAMAzxCwDAMGaK36q6oKpur6o7qh6eHi9c9rAAADCPWc/8/ockv9vdfyvJ65KcSHIoybHu3p/k2LQNAAC71pbxW1UvS/L3ktyaJN393e5+KsnNAauRFmcfcVkIo4g0nPc8kgAlLOiwiW9TUz+L3ZbvnMPgSdgUfy3yaDeTCBBe4vgMpPGgD1EuvIKyFrZPtYpg/GXPLr4VQ/ib6hsoh0vI62gbwLJzPX5hYEP3cnoEcx2Ot/PM8MG1HELO0LIRhIbNuNhFHoakq0ISdcNsyM/Tb0nhiQkH61l9Vx9xUL6YtJu1pyr66/jMm/Q+zPZz71D+NXVUFlsCcF5UvxFqV/ARZbMrRG1VAgT92Zc8/BgPS1Qk1cUWavW3wZm7JYg4PJlTN2oL+XoQlowP9MEHNb+WmX0ajSgqNQH/4xgTNlaaDZ0lRHFKBg8AcE+fwrqQIWVs9RY6q2D5PJKgORlWUcPLJcT/uBWPkG/Jo/tNFjYcdKddEoS2amtp0NDdTD3lIjwSfIZTC2C9Uu+kgXAYdnm0iw6srIx3mdGgTSvCHGsSc5mjXPNxBbfwu4PDJi/xQY0GGj5NRQPgeYVFS/zqvV3SF70Kp6cNJFNp8yx3DE2u3+UaeQNG0zo0Ml8Y

```
QAAeEGa6w5vAABwLhG/A6qq91bVyar67ClrF1XV3VX18PR44SpnZPeqqiuq6p6qOlFVD1XVO6Z1xxAzqaoXV9Unq+oz0zH0q9P6K6vqvukY+tD0y9RwRlV1XlU9UFUfn7Yd
Q2xJ/I7pfUmue97aoSTHunt/kmPTNpzOM0l+vrtfk+TqJD873drcMcSsvpPkmu5+XZIrk1xXVVcneVeSd0/H0DeS3LLCGTk3vCPJiVO2HUNsSfwOqLv/MMmfPW/5hiRHp+d
Hk9y4o0NxzujuJ7r709Pzp7P5L57L4hhiRr3pW9Pmi6afTnJNktundccQ31dVXZ7k+iTvmbYrjiFmIH55zqXd/USyGTdJLlnxPJwDqmpfktcnuS+OIc7C9L+rH0xyMsndSb
6Y5Knufmba5bFs/kcVnMlvJvnFJH8xbb88jiFmIH6Bbamqlyb5cJJ3dvc3Vz0P55bufra7r8zm3UGvSvKa0+22s1NxrqiqtyQ52d33n7p8ml0dQ3yPub7nlxeUJ6tqb3c/U
VV7s3k2Bk6rql6UzfB9f3d/ZFp2DHHWuvupqro3m9ePX1BVe6Yzd5cneXylw7GbvTHJW6vqzUlenORl2TwT7BhiS8788pw7kxyYnh9IcscKZ2EXm66ruzXJie7+jVNecgwx
k6paq6oLpucvSfLj2bx2/J4kb5t2cwxxRt39y919eXfvS3Jzkj/o7p+KY4gZuMnFgKrqA0l+LMnFSZ5M8q+TfCzJbUn+epKvJLmpu5//S//S3GQqvq7Sf4oyZ/kL6+1+5VsXvf
rGGJLVfW3s/nLSOdl8yTMbd39a1X1qiQfVcFU/luQXuvstjiFmIX4BABiGyx4AABiG+AUAYBjiFwCAYYhfAACGIX4BABiG+AUAYBjiFwCAYYhfAA
CG8f8A0Y8Gx4pkE+0AAAAASUVORK5CYII=\n",
      "text/plain": [
       "<Figure size 864x432 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "\n",
    "affordable_apps[cheap].hist(\"Price\", grid=False, figsize=(12,6))\n",
    "affordable_apps[reasonable].hist(\"Price\", grid=False, figsize=(12,6))\n"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Create a column in affordable_apps called affordability. It should have the value cheap if the price is lower than 5, and
reasonable otherwise"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 50,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0           cheap\n",
       "1           cheap\n",
       "2           cheap\n",
       "3           cheap\n",
       "4           cheap\n",
       "          ...    \n",
       "793         cheap\n",
       "794    reasonable\n",
       "795    reasonable\n",
       "796         cheap\n",
       "797         cheap\n",
       "Name: affordability, Length: 774, dtype: object"
      ]
     },
     "execution_count": 50,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps['affordability'] = np.where(affordable_apps['Price']< 5, 'cheap', 'reasonable' )\n",
    "affordable_apps['affordability'] \n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 55,
   "metadata": {},
   "outputs": [],
   "source": [
    "# or\n",
    "# affordable_apps[\"affordability\"] = affordable_apps['Price'].apply(lambda price :\"cheap\" if price <5 else
'reasonable')\n",
    "# affordable_apps[\"affordability\"] "
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Hint for pd.apply method asxi=\n",
    "axis{0 or ‘index’, 1 or ‘columns’}, default 0\n",
    "\n",
    "    Axis along which the function is applied:\n",
    "\n",
    "        0 or ‘index’: apply function to each column.\n",
    "\n",
    "        1 or ‘columns’: apply function to each row."
   ]
  },
{
 "cell_type": "code",
 "execution_count": 56,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "A    12\n",
     "B    27\n",
     "dtype: int64"
    ]
   },
   "execution_count": 56,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Explanation\n",
  "df = pd.DataFrame([[4, 9]] * 3, columns=['A', 'B'])\n",
  "s= df.apply(np.sum, axis=0)\n",
  "s\n"
 ]
},
{
 "cell_type": "code",
 "execution_count": 57,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0    13\n",
     "1    13\n",
     "2    13\n",
     "dtype: int64"
    ]
   },
   "execution_count": 57,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
    "ss = df.apply(np.sum, axis=1)\n",
    "ss"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "we saw that the reasonable apps are still somewhat skewed, although much less so than the dataset as a whole. We should be
mindful of this in our analysis moving forward.\n",
    "\n",
    "Having grasped the behavior of the price by itself, it's time to compare it to the other columns.\n",
    "\n",
    "Several columns stand out as being relevant. We'll focus on the rating, category and genres. Let's start by visualizing how
price relates to rating"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 58,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<matplotlib.axes._subplots.AxesSubplot at 0x7f96e5723350>"
      ]
     },
     "execution_count": 58,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYIAAAEGCAYAAABo25JHAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAAgAElEQVR4nO3df5QcdZnv8fdnJkMCJAgmI7BJMOEkugYIEWb5lTWXH64LiAn3EjXeE1k8y2FVV
MTdBVnvYVfU3Qt3LyjiXWTRIz90AYlC5IIiBhZQiXTYSfiRReaSYIKQDEMIiSSTmcxz/+jupKdT3dM93VX1na7ndU5Dd3XN1JNvVc+3q556vl+ZGc4557KrLe0AnHPOpcs7
AuecyzjvCJxzLuO8I3DOuYzzjsA55zJuXNoB1GvKlCk2Y8aMtMNwzrkxZdWqVa+ZWWfUe2OuI5gxYwa5XC7tMJxzbkyR9FKl9/zSkHPOZZx3BM45l3HeETjnXMZ5R+Cccxn
nHYFzzmWcdwTOOZdxsXYEktZLelpSt6R97vlU3vWSeiStkXRcnPF89vYnOerKB/js7U/GuZm65db1ce2Dz5Nb19eU39e3vZ/VG96gb3t/UHG1up5N27g7t4GeTduatg9alb
dPfeL+LCrOYaglrQe6zOy1Cu+fDXwWOBs4EfiGmZ1Y7Xd2dXXZaOoIZnzx/+6zbP3//GDdv6fZlt78BI/37N2575s1mdsuPGnUv+/e7pe5fNkaOtraGBga4prz5rJw3tTU4
2p1V97zNLc+8bs9r9vbxAEd7Q3tg1bVrGM0K5r1WZS0ysy6ot5L+9LQIuBWy3sCOFjS4c3eSKUzgLTPDHLr+obtYIDHevpG3be/n8mVr2DkwxLb+QXYODHHZsjV1f+tq
dlytrmfTtmGdAMDuIWtoH7SqZh2jWZHUZzHujsCAByWtknRRxPtTgQ0lrzcWlg0j6SJJOUm53t7euoO4/9lNdS2Py6NrX6treVwefeyXm5XC7tMJxzbkyR9FKl9/zSkHPOZZx3BM45l3HeETjnXMZ5R+Cccxn
8UfX90eyDVtWsYzQrkvosxt0RzDez44CzgIslLSh7XxE/s8+1KjO7ycy6tV1o+kmmH7M/A0NCwZQNDQ0w7ZP9U42p186YfXPX90eyDVt
WsYzQrkvosxtoRmNnvC//fDPwYOKFslY3A9JLX04DfNzu0by79k7qWJ6Vr5mTeN2vysGXvmzWZrpmTK/xEdZMnjuea8+YyoaONSePMaGjjWvOm8vkieNTjavVzTp0Eueff
```

```
MSwZe1tamgftKriMbpfuxg/ro392uXtU0VSn8XYBp2TdCDQZmbbCs8/AFxVttpy4DOS7iCfLN5qZq/EEc83lszjkju697y+fsm8ODZTt9suPIncuj4efeE1Fsye0vAOXjhv
KvNnTWHjlh1MO2T/UX/AFndN54kX+0ACMz7cNX3kH8qw49/5du58ciMif0p75YfmcPQfva2hfdCqcutfZ9duo3jyn3vpdU8WV7Hns1g4uuL4LMZ215CkI8mfBUC+w/mBmX1
N0icBzOxGSQJuAM4E3gI+YWZVbwkazV1Dfdv7mX/1CnYO7D0lndDRxi8vP90/pBG8verj7VW7nk3beP91j+6z/KFLFzDr0EkpRBS2Zh5b1e4aiu2MwMxeBI6NWH5jyXMDLo
4rhqJigmonexuzmKDyD+q+vL3q4+1Vu0qJ9e4Nb3hHECGpYyvt20cT4Qmq+nh71cfbq3aVEusjJdyzKqljKxMdQTFBNU7QLhgngkpQ3f6rdXz4xl9x+6/WpR0KEH57/eK5V
7n87tX84rlX0w4F2Nte48e17XmE1F6lFc9pi0qsn3/yEcGcDYTUVrD32Opog/Y26GiL57M45mYoG61/vP85BkvSIf90/3NBJKiO/YefsnXnbgCeXL+F//Xg86z+hzNTjgpu
ePiFYe31rYdfCKK9PnDdI/x20x8AuDO3kXcfeiA/u/TUdIMinwDtH9z7zS2UBGh5xfP5Jx/BVYuOSTEiIm4QD0OQbUX+s1hMEewmns9iJs4I7nlqA6++uWvYslfe3MU9T22
o8BPJuP1X6/Z0AkVbd+5O/czgF8+9uuePbdHzm/6Q+jfwUOOKqiy+9de/S/1bZYhxhRgThBtXUsd8JjqC+56ObrRKy5Ny75roO2UrLU/Kg89tqmt5UkKNq1oCNE0hxhViTN
W2n3ZcSR3zmegIzjnmsLqWJ2XR3OhhlSotT8oH5hxa1/Kk1BJXGqNahpoADSGu8v0RQkz1bD/tuJL6LGaiIzj3uOn7/EPbCsvTtPSUmbxtQvuwZW+b0M7SU2amFFHeGXOiO
8hKy5NyxpzDIturGNe93S8z/+oVLL15JfOvXsHy7pcTiatSojPtBGjaidmo/THr0El0lA0s06Ew2iqqgjftuJL6LGaiI/jFc68yVLZsqLA8bav/4Uy+unAOfzLjEL66cE4Q
ieJvP/xCXcuT0rNpW2ROpTj+f1qjWl5w86/rWp6kqxYdw0OXLuCfF8/loUsXJJb8rLQ/vv3wCwyUJYsHjNTzYn3b+3nypS3Dlj350pbUR0VN6rOYiY4g1GvLRUtPmckPP3l
K6mcCRfdUyFFUWp6Uatdx0xzV8lfrXq9redJmHTqJxV3TE/12W2l/VDqG0s6LhToqalKfxUx0BKFe8w7VuRVyFJWWJ6Xaddw0i7pOmfn2upZnQaX9UekYSjsvFmpRYFKfxU
x0BGfMOYx3H3rgsGXvPvTA1K95F4VWIPVXp81m/3HDL+TuP0781WmzU4oor9o172aNvDoa37vw5LqWJy2tIqmLT53F+HHD98dfnTY79rzYaG4YKC0KPGC/9mCKApP6LGamo
Oz5iHtxQxBqgdTgbqv6Oi35URj3Wlnyulkjr7aSNIqkSqeiBOOiBUfy3088Ys/+2DU4/Jt3+etmbbveaTCt+F8TwVa9xSQTZwQXfW9lXcuTEmqB1HU/WxuZ0LvuZ2vTCaig
lvaaPHE8x04/ONFO4Nxv/ntdy5OSRpFUeZK4f9D41iM9e97/9sMvsGNw+MG1Y9Cakvxs5IaB4s/2DxpvDeymf9CCmEIzzvYqlYmO4NGe6Pk9Ky1PSqhJ7HvXRHdElZYnJdT
2WvPy9rqWJyWNIqmRkq5xJj8bSfh6sjgDFsyKnuyl0vKkhJrEXjQ3OndSaXlSQm2vuVMn1rU8KWkUSY2UdI0z+dlIwteTxTGT1C7pPyTdF/HeBZJ6JXUXHhfGEcNNF5xY1/
Kk1JLEHk3iq9Hq2vPnH0l7RNHPpX/+nlH9vmYJMenft72fL5+7z7QbANzz2f+ScDTDpVFQNlLSPs7kZyM3DOxNFosDOtoZPy6MKTRbKVl8CbAWOKjC+3ea2WcSiGMfy7tfT
n2EyBNnTh6WuD7xyL1nKaNJfDWSLCv9+fLc8JRJ+9X8O+L0s0tP5RfPvcqDz23iA3MOTbUTGJ4UDdNVi47h/JNm0L3hDeZNPziRWoI0k/aNbDt/yKswI6Sqr5yg/rIcQfnr
Zoj1CJY0DfggcHOc2xnJ39z5VOTyS+/sTjUZVC2ZN5rEV6PVtaU/Xy6E0VqLzphzGFcvPjb1M4HSto7y2dufTDiqaGkUlFVK2ieR/BzNDQN7k8VDvLVrN/2DyVWmV/O1nzw
TOSrC137yTFO3E/dXma8Dl8E+/5ZS50laI+luSZGD/0i6SFJOUq63t7fuIH6+dnPk8t1GqsmgZlfKNprwivr5UmmP1hqSkdoKYMVvX0somrEj1Kr1UJPF9z1TYeTkCstHK7
aOQNI5wGYzW1VltZ8AM8xsLvAQcEvUSmZ2k5l1mVlXZ2dn3bH82XveEbm8XaSaDGp2pWyjCa+ony+V9mitIRmprQBOf9eUhKIZO0KtWg81WXzO0RVGTq6wfLTiPCOYDyyUt
B64Azhd0u2lK5hZn5kVz73+FTg+jkD++aPHRS6/7qPzUk0GNatStpgcBhqqri3dZrl3TOpIfbTWojSGmi5Xvn+ifHPpnwDpx5v29kuFWrUe6nSjX/rQ0ZEjJ3/pQ0c3dTux
JYvN7ArgCgBJpwJ/Y2ZLS9eRdLiZFc8JF5JPKmfKi73DC6TWlbyuJfEVlRz+5eWnjzpRV9xm11cfGlZbuXnbQBDJ9UaT4c1Uun8WfeuXkeukHW/a24+SRPJzNEKdbnT8OA3
Lq4wf1/xEduK3O0i6StLCwsvPSXpW0mrgc8AFcWzzf/xodeTyS+5IN1mcW9fH42VFbY/19JFbt3dZtcRXpeQw0FB17Y0PvxBZYJ92e6U51HQlkyeO50vLuiPfO/Pah1ONN8
T2Sir5Wa9Qp6psqcpiM3vEzM4pPL/SzJYXnl9hZkeZ2bFmdpqZ/Wcc23/g2ejKUyPdZPGjL0QnEystLxdXgqtSIirt9go1offsq9HjVv3n5rdSjTfE9koq+VmvUKeq9MriJ
jrrqOjKU5FusnjB7OhkYqXl5eJKcFVKRKXdXqEm9I467MDI5X/8jgNSjTfE9koq+VmvUKeqbJnK4hB89b9FV35+Y0m6yeKumZMjp8frmlnb0BdxDb0claCC9Nur+O/taIOO
NtHRRhAJvfs+f2rk8p9+4bTUhsaG+I6PRiSV/KxX2tN6VtJKlcXBCiERNHPKgTxWkieY2Rn97bKSuKo433HQfrz65q49rw+d1BFEe92V20C+3i1/3fSHuQ1BxFVJ2kNjp73
9KJMmtA+bcnRS2fwEaUmjCrsWx894+7BcYteM5k94lIkzgkrJ4krLk9KsBFWzh16+56kNwzoBgE3bBlKvLK4luZ6GkeYsTmNo7FJpb7/U7b9aFznvdNpzFhelUYVdTVLHfC
Y6gkrJ4krLkxJqgqpSBXHalcWNJtfjEvqcxSGpNDdx2nMWhyqpYz4THUGlZHGl5UkJNUFVqYI47criRpPrcfE5i2tXaW7itOcsDlVSx3wmOoJKyeJKyytpdoVmVILqI13TI
k9Lq2272XGde9x0Dj9o+Gijhx+0X+qVxY0m15shqq0rzU38v5dEV7Rn2dJTZsY+Z3ErSeqYz3SyuB5xVWiWz8F7V24jfzpryrDfXW3bccX1xbPn8Nd3dmPkbxu94uw5Df/O
ZkizBrVaW3eIfab3nH/1iiAqeUOzrSxHUP7aDbdp285hrzeXvW6GTJwRNDpncVwVmlFz8AJ8oWR47Grbjiuu4u8dtPwIrYNG6hWpkG6yuFpbR83xDARRyRuaUCuLQ5XUvOa
Z6AganbM4rgrNSnPtllbwVtt2XHGFWJEK6SaLq7VJtbmcQ2i3kIRaWRyqpObpzkRH0OicxXFVaFaaa7e0grfatuOKK8SKVEg3WVytTarN5RxCu4Uk1MriUCU1T3cmOoJG5y
yupUJzNAnbqDl4Aa4tGR672rbjqhwNdUjeNJPF1dr60j9/Dx0RA0KGUMkLYQ1DHWplcVGSbVXLtpKapzszyeLDyiply++KGUm1Cs1GErafPm32nqSsGXzlvx69z89W23Zcl
aOhDsm7uGt6PsEugRkf7kruTqZqbT3zHQcOu5Z7xCH78+OL56feCYQ4DPV1S+Zx6R3dDJHvBL6+ZF6q8RQl2Vb1bOvVrTurvm6GTJwRRFXKjmYO3qgKzUYStuVJ2SHgK/c9
F/mz1apDm105GuqQvMX2GhiCgd3GwFDySeyoto5K6P1uyw66f7clsbiihDgMdTGm4leMIcK4ESHJtqpnW0lVYmeiI4izUraRxGqoSdlQK55Dba+kEnr1CrG9QowJko2rnm0
lVYmdiY4gzkrZRhKroSZlQ614DrW9kkro1SvE9goxJkg2rnq2lVQlduwdgaR2Sf8h6b6I98ZLulNSj6SVkmbEEUOclbKNJGz3JmXFAR3tjB+nIJKLoQ7JG+KwypBcQq9eIb
ZXMab92sX4cW3s1x7GMZ9kW9WzraQqsZNIFl9Cfi7igyLe+0tgi5nNkrQEuBr4aBxBvLVrd9XXjWgkYZuvQ1L+nlFr/lyko1Ve8bzyxXRH+CwKcVhlgBNnTub5kjzBiUcmN
+xFNSG2V2796+zabRSP/lBuREiyrerZ1qJ5U4fl7Ba9t/ltFesZgaRpwAeBmyussgi4pfD8buAMSU3/a5hEwmU0Cdti0qh/cIi3du2mfzD9ZB4kV804WiENqwzhJteLQmov
b6v6tpVUe8V9aejrwGWwT1V50VRgA4CZDQJbgX2+Skm6SFJOUq63t7fuIEId+jbUxFmoyc9QhZpcD5G3VX2Saq/YOgJJ5wCbzWxVtdUilu0zaouZ3WRmXWbW1dnZWXcsoQ5
9G2riLNTkZ6hCTa6HyNuqPkm1V5xnBPOBhZLWA3cAp0u6vWydjcB0AEnjgLcBTZ/NI9Shb0NM5kG4yc9QxZ1cD6kyuFGh3ogQqqTaK7ZksZldAVwBIOlU4G/MbGnZasuBvw
B+DSwGVphZLCMNX3XuMVx6x95hlb9y7jFxbKZuISbzADonTRiW/HzHpAkpRhO+uJLrIVYGNyrUuYFDlcSNG4nXEUi6StLCwsvvAJMl9QBfAL4YxzZLqxmNcKoZi0JK5kG4c
wOHKq7keoiVwc0S2tzAoWqpYajN7BEzO6fw/EozW154vtPMPmxms8zsBDN7MY7th5qUDVWocwOHKq7kuh+3zoehbqJQk7KhCnVu4FDFlVz349b5MNRNFGo1Y6hCmBu4mtCS
p3El1OO9mWC0ejZt4+7cBno2bQtuH8ZttP9eH4a6yUKtZgzV4q7prFy39wauJId7ribU5GlccymHejNBva685+lhhVHtbflhVULah3Fp9Jj99Gmz+dsfrsYsP/r6xafNbnq
MmTgjCL2aMTTFJOWu3bbnEUKSMtTkadwJvdBuJqhX1Odv95AFtQ/j0ugxW/pZHBiK77OYiY7AqxnrE2qSMtS4vBK7upE+ZyHsw7g0eswmdcxnoiPwasb6hJqkDDUur8Subq
TPWQj7MC6NHrNJHfOZ6Ai8mrE+oQ6PHWpcXoldXdTnr71NLZEAH0mjCf+kbnTJTLL4xd7h13DXlb12w4U6PHaocXkldnXl1cSHHLjfmE+A16rRhH8SN7pk4ozAK2XrE+rw2
KHG5cdXbUqricd6Arxeo/33tsow1EHwStn6hJqUDTUuP75cXMb8MNQh8UrZ+oSalA01Lj++XFxaYRjqYHilbH1CrWjdmyxu2/MIIa7Qj68QhXbMh2rMD0MdtsuPIncuj4e
```

feE1FsyeEsyHNNRK2VArWnPrX6d/cO9ZQSgV4ou7pueHC5bALJhK7BCFesyHKolhuxXT8P+x6erqslwul3YYTdG3vZ/5V69g58DeP2wTOtr45eWnB/OHNyQ9m7bx/use3Wf
5Q5cuSPVWYN+PtfO2So+kVWbWFfVeJi4NhSrU5GeoQq0Q9/1YO2+rMMU5Z/EESb+RtFrSs5K+HLHOBZJ6JXUXHhfGFQ/kb/O79sHng7mtL9TkZ6hCrRD3/Vg7b6vRiTunEu
cZQT9wupkdC8wDzpR0UsR6d5rZvMLj5riCWXrzEyz+9hNcv6KHxd9+go/f/ERcm6rZ5Inj+cjx04Yt+0jXND9FriDUCvHJE8fzkS7fj7UI9UaEkN3b/TLzr17B0ptXMv/qF
Szvfrnp24hzzmIDthdedhQeqSQkqhX8pJk07tvez12rNg5bdlduI5ec8S7/YFQQ4ny3fdv7uSvn+7FWod6IEKLS0Ut3kj+TumzZGubPmtLUdos1RyCpXVI3sBn4uZmtjFjt
PElrJN0tKfJWC0kXScpJyvX29tYdR6gFP369dHRCm+/W92P9slZZPFotMfqome02s3nANOAESUeXrfITYIaZzQUeAm6p8HtuMrMuM+vq7OysO45QC378emlr8P3o4tJSo4+
a2RvAI8CZZcv7zKyY/fhX4Pg4th9qwU/o10vHUtFPmrGGWujm4pHksZbU34jYcgSSOoEBM3tD0v7A+4Gry9Y53MxeKbxcCKyNK549BT/5YSuDKfgJ9XrpWCr6CSHWUAvdXH
Olcawl8TcizjOCw4GHJa0BniSfI7hP0lWSFhbW+Vzh1tLVwOeAC+IIpJhwGRiCgSFjYIggRq0sCu16aahTQkYJIVafCjUb0jzW4v4bEeddQ2uA90Ysv7Lk+RXAFXHFUFRMu
BSz7rA34RLKH9+QjKX2CiHWaoVuoSS0XeNCONbikonKYk/m1WcstVcIsYZa6OaaK4RjLS6Z6AhCT8qGpjT5ecB+7UEnP0PYt6EWurn6VUsEp3msxZ2gzszoo6EmZUNlxf+a
SKkOsGYh7NsQC91cfWpJBKdxrCWRoPbRR90+fIRIlzWhHvPNjMtHH3V18UpZlzWhHvMtUVnsxqZWToo5FyXUYz6oymJJ10c8viJpUVOjidlYqpRNUwgJWNfaQvsshnrMJxV
XTTkCSTcBfwz8sLDoPOBZYDrwopl9vqlRVTHaHEEI1adjTd/2fk+uu6YL+bMY6jHfjLiq5Qhq7QhWAB8ws8HC63HAg8CfAU+b2ZxRRTYKo+kIQk0EOZc1/llMTzOSxVOBA0
teHwj8kZntJj8BTdBCTQQ5lzX+WQxTrXUE1wDdkh4hP2rbAuAfJR1IfvjooIWaCHIua/yzGKaazgjM7DvAKcA9hcefmtnNZvYHM/vbOANshmLCZb92GN/exn7tBJEICl1oC
T039oWalM26eiqL24Dews/MkjTLzB6NJ6zmy61/nV27gcKAUT5McHUhJ/Tc2BZCJbgbrqaOQNLVwEfJ3ylUPK8zYEx0BJWGCT7/pBk+FECEpOZJddk1eeJ4P5YCUusZwbnA
u0tmExtTfJjg+rTycLvOuX3VetfQi0BHnIHEyYcJro8n9JzLllo7grfI3zX07dLq4mo/IGmCpN9IWl2YhezLEeuMl3SnpB5JKyXNqP+fMDIfJrg+Y2kY6pDk1vVx7YPPk1v
Xl3Yow3jSf+zr2bSNu3MbYpv1rtZLQ8sLj3r0A6eb2XZJHcDjkh4wsydK1vlLYIuZzZK0hPycxh+tczs1Of6db+eO3/wO0YYxRNc73x7HZlrGWBqGOgRLb36Cx3vyHcD1K3
p436zJ3HbhSSlH5Un/VnDlPU8Py3Gef/IRXLXomKZuo9bbR2+JeozwM2Zm2wsvOwqP8r8oi4Di77kbOEOS6oi/JsXk567d0L97iF27w5qzODTF9uofNN4a2E3/oHl7VZFb1
7enEyh6rKcv9TODEOZzdo1Jaj7sqh2BpLsK/39a0pryx0i/XFK7pG5gM/nJ61eWrTIV2ABQGL5iKzA54vdcJCknKdfb21vbv6yEVzPWx9urPo++8Fpdy5Pi+3Hsq3ajSzON
dGnoksL/zxnNLy8MQTFP0sHAjyUdbWbPlKwS9e1/n+sQZnYTcBPkxxqqNw5PftbH26s+C2ZP4foVPZHL0+T7cexL6kaXqmcEZvZK4emnzeyl0gfw6Vo3YmZvAI8AZ5a9tZH
8CKbFgezeBrxe6++tlVcz1sfbqz5dMyfzvlnDT2TfN2syXTP3OblNlO/HsS+pG11qTRb/GXB52bKzIpbtIakTGDCzNyTtD7yffDK41HLgL4BfA4uBFRbT3JlezVgfb6/63H
bhSeTW9fHoC6+xYPaU1DuBIt+PY9+LvX8Y9npd2etmqNoRSPoU+W/+R5blBCYBvxzhdx8O3CKpnfyZx11mdp+kq4CcmS0HvgPcJqmH/JnAklH+O2ri1Yz18faqT9fM9M8Co
vh+HLuq3YjQzGNtpDOCHwAPAP8EfLFk+TYzq3oJx8zWAO+NWH5lyfzOIzewIRsoRbDWz9Wb2sUJeYAf5ZO5ESUdU+1nnnHONqXTDQbNvRKh1zuIPSXoB
WAf8O7Ce/JnCmBJ3dZ5zzjVTUjci1Jos/ipwEvCQmb1X0mnAx5oaScyqSqM5zzrlmW9w1nZXrtpC/GCM+3DW96duodayhATPrA9oktZnZw8C8pkcTk6Sq85xzrpn2joowxK7
dxq7d8VSH19oRvCFpIvn5B74v6RvAYFMjiVFS1XnOOddMSVWH19oRLCI/AumlwE+B/wd8qKmRxMiHoXbOjUVJVYfXOujcH8xsyMwGC4PNfYt9q4SD5cNQuyT4cM+u2ZIaEn
6kgrKDgIvJDw63HPh54fXfAt3A95saTYyuWnQM5580g+4NbzBv+sHeCbim8uGeXVySGBJe1UZ0kHQvsIX8EBBnAIcA+wGXmFl3LBGNoKury3K5XBqbdi5S3/Z+5l+9gp0De
0/hJ3S08cvLT/eKXteQZh5bklaZWVfUeyPdPnqkmR1T+CU3A68BR5iZ327jXIHP8eziktSxNVKOYKD4pDCk9DrvBJwbzod7dnEJJVl8rKQ3C49twNzic0lvNjUS58YoH+7Z
xSWpY6tqjiBEniNwoerb3u/DPbtYNOPYaiRH4JyrkQ/37OIS97FVa0GZc865FuUdgXPOZVxsHYGk6ZIelrRW0rOSLolY51RRsMSZIxgE/trMnpI
0CVgl6edm9lzZeo+Z2TkxxuGcC4hXYYcntjMCM3vFzJ4qPN8GrCU/ViVIzLqOKwyrvHBhiW/8gOwfiGVbZ1SeRHIGkGeTnL14Z8fbJklZLekDSURV+/iJJOUm53t7eGCN1zs
UpqWGVXX1i7wgK8xgsAz5vZuVFaE8B7zSzY4FvAvdE/Q4zu8nMusys69WAEUjqIN8JfN/MlbbC8/vBzokNXdWZjdqntBzzbZ3WGVxQEc74
8fJq7ADEFuyWJKA7wBrzezaCusmOqS+umFztPKHn4pIfy0AgCkMru7TFedfQfODjwNOSikNVfNkNW/x1wBICZ3QgsBj4laRDYASyxsTbmRQsqTegVRz28bNka5s+a
4t/cXEOKx1b/4N7LQ35spS+2jsDMHiff51db5wbghrhicKPyq7uPixFaZMVRaHes07tLg8oefi4sdWmDIz6Fyo17xDjKuY0LusLC7/xuYa5cdWmDIxDHWoUwmGGleRD6v
s4uLHVvIyPwx1qNclQ42ryIdVdnHxYyssmcgRhHpdMtS4nHPZkomOINSpBEONyzmXLZnIERSFel0y1Licc60j8zmColCvS4Yal3MuGzJxacg551xl3ppIs2QAAArhSURBVB
E451zGeUfgnHMZ5x2Bc85lnHcEzjmXcd4ROOdcxnlH4JxzGRdbRyBpuqSHJa2V9KykSyLWkaTrJfVIWiPpuLjicc65RoU2ZHyzxFlQNgj8tZk9JWkSsErSz83suZJ1zgJmF
x4nAv9S+L9zzgUlxCHjmyW2Mwze8XMnio83WasBcpbbCXkFwm83waGfwq+BcAqrR+Ws6fC4YnL0udEonb51W/8gOweGuGzZmpY5M0gkRyBpBvBeYEYGXZW1OBDSWvN7JvZ4GkiyTlJO
CtM55yIVh4wvVRyvhXE3hFImggsAz5vZuVFaE8B7zSzY4FvAvdE/Q4zu8nMusys69WAEUjqIN8JfN/MfhSxykZgesnracCmuz4zJOefq1epDxseWLJYk4DvAWjO
7tsJqy4HPSLqDfJJ4k3WgvnTWX+rCktOWR8nHcNzQc+Djwtqbuw7O+AIwDM7EbgdM7C/hEjPE451xdG5pXFzj
mXcd4RBKBVqxdWfEqmzFamC4Ys7I94mbR6ncJmUZNT8KQ2e60Yf0ZUvr5RVVjQx2adqNzrdyYyKPFkccpauVWrOpp1Ad/DirlXmgfhl3t1ArQ7kjZUeTLM4l3rxKeSb1cr
EN5d8d6iN9L5Qh0rWqpEgKpqanYt1TVLl6XMS4C8ZU6l8UDlEKBmdzZOuafSn8BrbMZlfD4hwecbnwWzQMiItsuFSQyYXR8Px2dlk7
dnZ2JBeicc1mQWkdgZm+a2fbC8/uBDklT0orHOeeyKrWOQNJhkR4fkIhlr604nHOuayK7a4hSf8GnApMkbQR+A8DMbgQWA5+SNAjsAJbYWBvzwgWmjU9sfp4t1gvwr
+B/O2lzjnnUpT2XUPOOedS5h2Bc85lnHcEzjmXcd4ROOdcxnlH4JxzGecdgXPOZZx3BM45l3HeETjnXMZ5R+CccxnnHYFzzmWcdwTOOZdx3hE451zGeUfgnHMZ5x2Bc85ln
HcEzjmXcbF1BJK+K2vqkbRG0GcqvC3/9aD1b/vu1O5l/vVoEA=LAbf8r7b+xuHmzw0BbmH3qeGq+3mEzznhQ+N4OS97fvMvH+pFxfkkB5L0W4A7b98jJYO9Wn2zGpnMh5b/7GCgvVC1ZjJ0N5+7Jx2NBgFcEIb5p8L9
DsvVyfejG8tiOyMws0eB16ussgi41ffKeAA6WdHhc8YSob3s/ly9bw86BIbb1D7JzYIjLlq3xb5RjjO9HN9almSOYCmwoeb2xsGwfki6SlJOU6+3tTSS4JGzcsoOOtuG7oKO
tjY1bdqQUkRsN349urEuzI1DEssjJ683sJjPrMrOuzs7OmMNKzrRD9mdadGaGjYsoGhIaYdsn9KEbnR8P3oxro0O4KNwPSS19OA36cUSyomTxzPNefNZUJHG5PGj2NCRxvXnD
eXyRPHpx2aq4PvRzfWxZksHsly4DOS7iCfJN5qZq+kGE8qFs6ZyN5GF1s6byvxZU9i4ZQfTDtnf/3iMUb4f3VgWW0cg6d+AU4EpkjYCvwEdMc4R8NAJjZjcD9w9wNlAD/AW8Im4
/D96Maq2DoCM/vYCO8b8Jm42h9PJk5oZ8wy1ezmrM847AOecyzjsC55zLOO8InHMu47wjcM65jPOOwDnnMs47Auecyzjv8m7LBNZ33j2tzMu47wjcM65jPOOwDnnMs47Auecyz
NIOknJl1pR1HOY+rdiHBB5XvTyu+sQVl18acs65jPOOwDnnMi6rHcFNaQdQgcdVuxBjgjDj8rJK5M5Auecc3tl9YzAOedcwdgRycS3bEUj6rqTnTNkp6p8L4kXS+

```
pR9IaSccFEtepkrZK6i48rkwgpumSHpa0VtKzki6JWCfx9qoxrjTaa4Kk30haXYjryxHrjJd0Z6G9VkqaEUhcF0jqLWmvC+OOq2Tb7ZL+Q9J9Ee8l3l41xpVKe0laL+npwj
ZzEe839/NoZi35ABYAxwHPVHj/bOAB8lNmngSsDCSuU4H7Em6rw4HjCs8nAb8F5qTdXjXGlUZ7CZhYeN4BrAROKlvn08CNhedLgDsDiesC4IYk26tk218AfhC1v9JorxrjS
qW9gPXAlCrvN/Xz2LJnBGb2KPB6lVUWAbda3hPAwZIODyCuxJnZK2b2VOH5NmAtMLVstcTbq8a4Eldog+2Flx2FR/ldF4uAWwrP7wbOkBQ1T3fScaVC0jTgg8DNFVZJvL1q
jCtUTf08tmxHUIOpwIaS1xsJ4I9MwcmF0/sHJB2V5IYLp+TvJf9tslSq7VUlLkihvQqXE7qBzcDPzaxie5nZILAVmBxAXADnFS4n3C1pesT7cfg6cBkwVOH9VNqrhrggnfY
y4EFJqyRdFPF+Uz+PWe4Ior5thPDt6SnyY4IcC3wTuCepDUuaCCwDPm9mb5a/HfEjibTXCHGl0l5mttvM5gHTgBMkHV22SirtVUNcPwFmmNlc4CH2fguPjaRzgM1mtqraah
HLYm2vGuNKvL0K5pvZccBZwMWSFpS939T2ynJHsBEo7d2nAb9PKZY9zOzN4um9md0PdEiaEvd2JXWQ/2P7fTP7UcQqqbTXSHGl1V4l238DeAQ4s+ytPe0laRzwNhK8JFgpL
jPrM7P+wst/BY5PIJz5wEJJ64E7gNMl3V62ThrtNWJcKbUXZvb7wv83Az8GTihbpamfxyx3BMuB8wvZ95OArWb2StpBSTqseG1U0gnk91FfzNsU8B1grZldW2G1xNurlrhS
aq9OSQcXnu8PvB/4z7LVlgN/UXi+GFhhhSxfmnGVXUdeSD7vEiszu8LMppnZDPKJ4BVmtrRstcTbq5a40mgvSQdKmlR8DnwAKL/LsKmfx9gmr0+bpH8jf0fJFEkbgb8nnzz
DzG4E7iefee8B3gI+EUhci4FPSRoEdgBL4v5AkP9m9HHg6cL1ZYC/A44oiSuN9qolrjTa63DgFknt5Dueu8zsPklXATkzW06+A7tNUg/5b7ZLYo6p1rg+J2khMFiI64IE4o
oUQHvVElca7XUo8OPC95txwA/M7KeSPgnxfB59iAnnnMu4LF8acs45h3cEzjmXed4ROOdcxnlH4JxzGecdgXPOZZx3BM5VIWl3YQTIZyT9UNIBFda7v3gPv3Njjd8+6lwVk
rab2cTC8+8Dq0qL2wrFbDKzamPVOBc0PyNwrnaPAbMkzVB+joT/Q36so+nKjx8/BUDS+YVBylZLuq2wrFPSMklPFh7zU/x3ODdMy1YWO9dMhfFvzgJ+Wlj0buATZvbpwvvF
9Y4CvkR+0LDXJL29sP43gOvM7HFJRwA/A96T4D/BuYq8I3Cuuv1Lhrd4jPxQCH8EvFQYB77c6cDdZvYagJkVB057PzBHe4fYP0jSpMI8C86lyjsC56rbURjWeY/CH/M/VFh
fRA8H3AacbGY7mhuec43zHIFzzfUL4COSJgOUXBp6EPhMcSVJ8yJ+1rlUeEfgXBOZ2bPA14B/l7QaKN5h9Dmgq5BEfg74ZFoxOlfObx91zrmM8zMC55zLOO8InHMu47wjcM
65jPOOwDnnMs47AuecyzjvCJxzLuO8I3DOuYz7/9A7+dDgM1P0AAAAAElFTkSuQmCC\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "affordable_apps[cheap].plot(kind = 'scatter', x =\"Price\", y = \"Rating\")\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Measuring \"how related\" two numerical ['Rating','Price'] variables."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 59,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
      "    .dataframe tbody tr th:only-of-type {\n",
      "        vertical-align: middle;\n",
      "    }\n",
      "\n",
      "    .dataframe tbody tr th {\n",
      "        vertical-align: top;\n",
      "    }\n",
      "\n",
      "    .dataframe thead th {\n",
      "        text-align: right;\n",
      "    }\n",
      "</style>\n",
      "<table border=\"1\" class=\"dataframe\">\n",
      "  <thead>\n",
      "    <tr style=\"text-align: right;\">\n",
      "      <th></th>\n",
      "      <th>Rating</th>\n",
      "      <th>Reviews</th>\n",
      "      <th>Size</th>\n",
      "      <th>Price</th>\n",
      "    </tr>\n",
      "  </thead>\n",
      "  <tbody>\n",
      "    <tr>\n",
      "      <th>Rating</th>\n",
      "      <td>1.000000</td>\n",
      "      <td>0.101934</td>\n",
      "      <td>0.098124</td>\n",
      "      <td>-0.048762</td>\n",
      "    </tr>\n",
      "    <tr>\n",
      "      <th>Price</th>\n",
      "      <td>-0.048762</td>\n",
      "      <td>0.004918</td>\n",
      "      <td>0.078832</td>\n",
      "      <td>1.000000</td>\n",
      "    </tr>\n",
      "  </tbody>\n",
      "</table>\n",
      "</div>"
     ],
     "text/plain": [
      "          Rating   Reviews      Size     Price\n",
      "Rating  1.000000  0.101934  0.098124 -0.048762\n",
      "Price  -0.048762  0.004918  0.078832  1.000000"
      ]
     },
     "execution_count": 59,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps[cheap].corr().loc[['Rating','Price']]\n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "In the graph above, we see that there doesn't seem to be any clear relation between price and rating for the cheap apps. In fact, the Pearson coefficient in this instance is around -0.05:\n",
    "\n",
    "This is good news for our price tweaking strategy, because it suggests that we can change prices without it being reflected in the apps' rating.\n",
    "\n",
    "We can increase the price of those apps that cost less than, say, the mean cheap price, to the mean cheap price. We would then monitor the behavior of these apps – possibly by using statistical techniques like hypothesis testing, which you'll learn later – to confirm that there is an increase in profits."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 60,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Find the mean price of the cheap apps and assign it to cheap_mean.\n",
    "cheap_mean = affordable_apps.loc[cheap ,\"Price\"].mean()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 61,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "2.5973246329526987"
      ]
     },
     "execution_count": 61,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "cheap_mean "
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "For only the cheap apps, create a column in affordable_apps called price_criterion that takes the value 1 when the app's price is lower than cheap_mean, and 0 otherwise."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 62,
   "metadata": {},
   "outputs": [],
   "source": [
    "affordable_apps.loc[cheap, \"price_criterion\"] = affordable_apps[\"Price\"].apply(\n",
    "    lambda price: 1 if price < cheap_mean else 0\n",
    ")\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 63,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0      0.0\n",
       "1      0.0\n",
       "2      0.0\n",
       "3      0.0\n",
       "4      0.0\n",
       "      ... \n",
       "791    1.0\n",
       "792    1.0\n",
       "793    1.0\n",
       "796    1.0\n",
       "797    1.0\n",
       "Name: price_criterion, Length: 613, dtype: float64"
      ]
     },
     "execution_count": 63,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps.loc[cheap, \"price_criterion\"]"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Create a scatter plot for the reasonable "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 64,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<matplotlib.axes._subplots.AxesSubplot at 0x7f96e5b8a690>"
      ]
     },
     "execution_count": 64,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYIAAAEGCAYAAABo25JHAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYi
B2ZXJzaW9uMy4xLjMsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+AADFEAAAcUklEQVR4nO3dfZAc9X3n8fdnhSwpSDx4tcEESRG5pfJkhIL3OGQFCsuOC8eKlByBIxUO002WK+
IGKfXcYYqeKs1XnckEltuNznV0YLubBDsYQA6FsH9iYAOHBGYGQTMDHXoSRbCwtiwSSIxSJ/d4f0ytmRzM7vavpnu7pz6tqa2d7eme+/duH73TP9/v7KSIwM7PqGuh1AGZm
```

1ltOBGZmFedEYGZWcU4EZmYV50RgZlZxR/U6gJlavHhxLF++vNdhmJmVysaNG1+MiKFW95UuESxfvpxardbrMMzMSkXSj9vd50tDZmYV50RgZlZxTgRmZhXnRGBmVnFOBGZ
mFedEYGZWcZkmAknPSdoiaZOkw2o+Vfd5SaOSNks6Pct4rLrG9+7nyW27Gd+7v9eh9C2PcXnl0Ufwtoh4sc197wJOST7+A/DF5LNZ19y56Sdceftm5g4McGBigmvOW8G6lS
f1Oqy+4jEut15fGloP3Bh1jwLHSTqxxzFZHxnfu58rb9/Mqwcm2LP/IK8emOCK2zf7VWsXeYzLL+tEEMA9kjZKurTF/ScB2xq+3p5sm0LSpZJqkmpjY2MZhWr9aPuufcwdm
PprPndggO279vUoov7jMS6/rBPB6og4nfoloA9JOrvpfrX4nsOWTIuIayNiJCJGhoZaTpVh1tKS4xdwYGJiyrYDExMsOX5BjyLqPx7j8ss0EUTET5PPO4FvAmc07bIdWNrw
9RLgp1nGZNUyuHAe15y3gvlzB1g07yjmzx3gmvNWMLhwXq9D6xse4/LL7M1iSUcDAxGxJ7n9TmBD0253AZdJuoX6m8QvR8QLWcVk1bRu5UmsHl7M9l37WHL8Av+DyoDHuNy
yrBo6AfimpMnn+VpEfEfS+wEi4kvAt4DfBUaBfwX+JMN4rMIGF87zP6eMeYzLK7NEEBH/ApzWYvuXGm4H8KGsYjAzs856XT5qZmY95kRgZlZxTgRmZhXnRGBmVnFOBGZmFe
dEYGZWcU4EZmYV50TQQW3rOJ+550fUto5n+jyjO/ZwW20bozv2ZPo83VK2uefLNr5mecpjPYLSuui6R3lotJ4APn/fKGcND3LTJWd2/XmuumMLNz76/KGvL161jA3rT+368
3RL2eaeL9v4muXNZwRt1LaOH0oCkx4cHe/6mcHojj1T/kkB3PjI84V95Vq2uefLNr5mveBE0MYDz7ZeVK3d9tnatG33jLb3Wtnmni/b+Jr1ghNBG2efsnhG22dr5dLjZrS9
18o293zZxtesF5wI2hg5eZCzhgenbDtreJCRkwfbfMfsDJ+wiItXLZuy7eJVyxg+YVFXn6dbyjb3fNnG16wXVJ8AtDxGRkaiVqvl9ny1reM88OyLnH3K4q4ngUajO/awadt
uVi49rhT/pMb37i/V3PNlG1+zbpO0MSJGWt7nRGBm1v+mSwS+NGRmVnGVTgRpmqKa98mqkaofG7R6dUxlG0uzXqtsQ1mapqjmfS4YWcKtte1db6TqxwatXh1T2cbSrAgqeU
aQpimq1T43PvJ81xup+rFBq1fHVLaxNCuKSiaCNE1RrfZp1o1Gqn5s0OrVMZVtLM2KopKJIE1TVKt9mnWjkaofG7R6dUxlG0uzoqhkIkjTFNVqn4tXLet6I1U/Nmj16pjKN
pZmRVHpPoI0TVHN+2TVSNWPDVq9OqayjaVZHtxQZmZWcW4oo3Xd+x2Pb+OSG/6JOx7f1vb7ZtNr0K34spLXMXVL2eI1K5tK9BG0qnu/56mf8bNX/g2A7z69k6u/8wyPfPx3
pnzfbHoNZlO3nufCKXkdU1XjNSujvj8jaFf3PpkEJr3wyr9NOTOYba/BTOvW81w4Ja9jqmq8ZmXV94lgJguQ3L3lZ4duz7bXYKZ163kunJLXMXVL2eI1K6u+TwQzWYBk7al
vOnR7tr0GM61bz3PhlLyOqVvKFq9ZWfV9ImhX937iMW+Ysu3EY97A75++9NDXs+01mGndep4Lp+R1TFWN16ysKlM+2qru/Y7Ht3H3lp+x9tQ3TUkCjWbTazAbeS6cktcxdU
vZ4jUrIvcRmJlVXE/7CCTNkfSEpLtb3PdeSWOSNiUfl2QdT6M0tfu1reN85p4fUds6fkT7dHLzw1s5/0sPc/PDW2f9GN2KpR+Vrc/AP0fLU+ZnBJL+KzACHBMRa5vuey8wE
hGXpX28bp0RpKndv+i6R3lo9PU/xLOGB7npkjNnvE8np33iO7z86muHvj52/hye/MS5M3qMbsXSj8rWZ+Cfo2WhZ2cEkpYA7wauy/J5ZipN7X5t6/iUP0aAB0fHp7xCS7NP
Jzc/vHVKEgB4+dXXZnxm0I1Y+lHZ+gz8c7ReyPrS0OeAK4Dp5nM+T9JmSbdJavmOraRLJdUk1cbGxo44qDS1+w88+2LLfRq3p9mnkzs3vzCj7e10I5Z+VLY+A/8crRcySwS
S1gI7I2LjNLv9PbA8IlYA3wVuaLVTRFwbESMRMTI0NHTEsaWp3T/7lMUt92ncnmafTtavOHFG29vpRiz9qGx9Bv45Wi9keUawGlgn6TngFmCNpJsbd4iI8YiYPEf/MvCWDO
M5JE3t/sjJg5w1PDhln7OGBxk5eXBG+3Ry0VtP5tj5c6ZsO3b+HC5668mpH6NbsfSjsvUZ+OdovZBL+aikc4DLW7xZfGJEvJDc/gPgyoiY9l2xbpaPpqndr20d54FnX+TsU
xa3/WNMs08nNz+8lTs3v8D6FSfOOAl0O5Z+VLY+A/8crdt63kfQmAgkbQBqEXGXpE8D64CDwEvAByLimekey30EZmYz1/NE0E2zTQStXhGmedWV5pVkN7qC83zFWqR48+yo
Nquy6RJBJdYjaFVHfmtt26Eyvc/fN9qyVjtN/Xk31hLIs869SPHmuQ6DmbXX95POtaojv/wbmzvWaqepP+/GWgJ51rkXKd4812Ews+n1fSJoVUcOrS+HNdZqp6k/78ZaAnn
WuRcp3jzXYTCz6fV9ImhVRw5quW9jrXaa+vNurCWQZ517keLNcx0GM5te3yeCVnXkf3n+io612mnqz7uxlkCede5FijfPdRjMbHquGnLV0Kwew1VDZuXi8lEzs4rr6XoElm
4u/DRrI+QVT5p4yza/v5m1V4k+gl7KqxehW/Gkibds8/ub2fR8RpChvHoRuhVPmnjLNr+/mXXmRJChvHoRuhVPmnjLNr+/mXXmRJChvHoRuhVPmnjLNr+/mXXmRJChvHoRu
hVPmnjLNr+/mXXm8tEc5NWL0K140sRbtvn9zarOfQRmZhXnPoI2alvH+cw9P5oy62iz5nr52dTPp+kRyLOPwIqvjL8PZYw5D2XoualsH8FF1z064/UILnjLEm7duH1G9fNp
egQ8L781KuPvQxljzkNZem4qeUZQ2zo+q/UIbnz0+RnVz6fpEfC8/NaojL8PZYw5D2XqualkImhcd6Dd9tbrGEzVqX4+TY+A5+W3RmX8fShjzHkoU89NJRNB47oD7ba3Xsd
gqk7182l6BDwvvzUq4+9DGWPOQ5l6biqZCEZOHpzVegQXr1o2o/r5ND0CnpffGpXx96GMMeehTD03lS4fnc16BLOpn0/TI+B5+a1RGX8fyhhzHorSc+M+AjOzinMfgZmZte
VE0EEZmkEa5dXU4+Yha1a2vxV7XWUbytIoSzPIpLyaetw8ZM3K9rdiU/mMoI0yNYNAfk09bh6yZmX7W7HDORG0UaZmEMivqcfNQ9asbH8rdjgngjbK1AwC+TX1uHnImpXtb
8UO50TQRpmaQSC/ph43D1mzsv2t2OHcR9BBUZpB0sqrqcfNQ9asbH8rVeOGMjOziutpQ5mkOZKekHR3i/vmSfq6pFFJj0lannU81h3uIzDLV5Z9Gnn0EXwYeBo4psV97wN2
RcSwpAuBq4H/lENMdgTcR2CWr6z7NDI9I5C0BHg3cF2bXdYDNyS3bwPeLklZxmRHxn0EZvnKo08j60tDnwOuANpN7H8SsA0gIg4CLwOHTQMq6VJJNUm1sbGxrGK1FNxHYJa
vPPo0MksEktYCOyNi43S7tdh22LvXEXFtRIxExMjQ0FDXYrSZcx+BWb7y6NPI8oxgNbBO0nPALcAaSTc37bMdWAog6SjgWOClDGOyI+Q+ArN85dGnkap8VNLnW2x+GahFxJ
0pvv8c4PKIWNu0/UPAqRHx/uTN4v8YERdM91guHy0G9xGY5etI+zSmKx9NWzU0H/g14BvJ1+cBTwHvk/S2iPjIDILZQD2B3AVcD9wkaZT6mcCFaR/Hemv4hEVOAGY5Glw4L
7NGvbSJYBhYk7yhi6QvAvcAvwNs6fTNEXE/cH9y+6qG7a8C588o4i7KqxOyHzsu+/GYzKoqbSI4CTia+uUgktu/FBGvSSrlXLN5zZ/ej/O09+MxmVVZ2jeLrwE2SfobSV8B
ngD+UtLRwHezCi4rec2f3o/ztPfjMZlVXapEEBHXA28F7kg+fjsirouIn0fER7MMMAt5zZ/ej/O09+MxmVXdTMpHB4Ax6m/qDks6O5uQspfX/On9OE97Px6TWdWlSgSSrgb
+EfgL4KPJx+UZxpWpvOZP78d52vvxmMyqLm0fwY+AFRHR8wvB3ewjcNXQ7PXjMZn1s270EfwLMBfoeSLopizrcnvxPHnqx2Myq6q0ieBfqVcNfY+GZBARf5ZJVDlJ86rWZw
3ZqupxmxVJ2kRwV/LRN9LUwrvXIFtVPW6zoqnkUpXje/ez+ur7ePXA69Uv8+cO8I9Xrjn0qjTNPt2Q1/MUTVWP26xXZr1UpaRbk89bJG1u/sgi2DykqYV3r0G2qnrcZkXU6
dLQh5PPa6fdq2TS1MK71yBbVT1usyKa9owgIl5Ibn4wIn7c+AF8MPvwspGmFt69Btmq6nGbFVHaPoLHI+L0pm2bI2JFZpG1kXcfgauGslXV4zbL26z7CCR9gPor/19pek9g
EfVO41JLUwvvXoNsVfW4zYqk03sEXwO+DXwa+POG7XsiwktKtuFXuWZWJtMmgoh4mfoaBH8EIOkXqa9WtlDSwoh4PvsQy8W18WZWNmknnfs9Sc8CW4F/AJ6jfqZgDTxXv5m
VUdppqP8HcCbwfyPiZODt9MF7BN3m2ngzK6O0ieBARIwDA5IGIuL7wMoM4yol18abWRmlTQS7JS0EHgC+KumvgYPZhVVOro03szJKO+ncemAf8F+APwaOBTZkFVSZrVt5Eq
uHF7tqyMxKI1UiiIifJzcngBskzQEuBL6aVWBl5tp4MyuTTpPOHSPpY5K+IOmdqruM+kI1F+QTYvmN793Pk9t2H3H1UNEex8z6Q6czgpuAXcAjwCXU1yp+A7A+IjZlHFtf6
FZfQdEex8z6x7RzDUnaEhGnJrfnAC8CyyJiT07xHaabcw1lrVtz7hftccysfGa9HgFwYPJGRLwGbO1lEiibbvUVFO1xzKy/dLo0dJqkV5LbAhYkXwuIiDgm0+hKrlt9BUV7
HDPrL53WI5gTEcckH4si4qiG204CHXSrr6Boj2Nm/aWSaxbnrVuzkRbtccysPGa9HoF1R7f6Cor2OGbWH9JOMTFjkuZL+oGkJyU9JemTLfZ5r6QxSZuSj0uyisfMzFrL8ox
gP7AmIvZKmgs8JOnbEfFo035fj4jLMowjc77UUnz+GVXL6I49bNq2m5VLj2P4hEW9DqfwMksEUX/zYW/y5dzko1xvSKTgBq3i88+oWq66Yws3Pvr6mlkXr1rGhvWn9jCi4s
vs0hDUm9AkbQJ2AvdGxGMtdjtP0mZJt0lammU83eaFaIrPP6NqGd2xZ0oSALjxkecZ3eH2p+lkmggi4rWIWAksAc6Q9OamXf4eWB4RK4DvAje0ehxJl0qqSaqNjY1lGfKMu
EGr+PwzqpZN23bPaLvVZZoIJkXEbuB+4Nym7eMRMfnS7MvAW9p8/7URMRIRI0NDQ5nGOhNu0Co+/4yqZeXS42a03eqyrBoaknRccnsB8A7gmaZ9Tmz4ch3wdFbxZMENWsXn
n1G1DJ+wiItXLZuy7eJVy/yGcQeZNZRJWkH9Us8c6gnn1ojYIGkDUIuIuyR9mnoCOAi8BHwgIp5p+6AUs6HMFSnF559Rtbhq6HDTNZS5s9jMrAKOZPbRSin6gi1Fjq/IsVk

```
+Rnfs4bbaNlfolJCnmEgUvda8yPEVOTbLh2v3y81nBBS/1rzI8RU5NsuHa/fLz4mA4teaFzm+Isdm+XDtfvk5EVD8WvMix1fk2Cwfrt0vPycCil9rXuT4ihyb5cO1++Xn8t
EGRa81L3J8RY7N8uHa/WLzwjQpFX3BliLHV+TYLB/DJyxyAiipSl8aKlLte5FiMbNqqewZQZFq34sUi5lVTyXPCIpU+16kWMysmiqZCIpU+16kWMysmiqZCIpU+16kWMysm
iqZCIpU+16kWMysmirdR1Ck2vcixWJm/cd9BG0Uqfa9SLGYWbVU8tKQmZm9zonAzKzinAjMzCrOicDMrOKcCMzMKs6JwMys4pwIzMwqzonAzKzinAhy4LUGzKzIKt1ZnAev
NWBmReczggx5rQEzKwMnggx5rQEzKwMnggx5rQEzKwMnggx5rQEzKwO/WZyxdStPYvXwYq81YGaF5USQA681YGZFltmlIUnzJf1A0pOSnpL0yRb7zJP0dUmjkh6TtDyreMz
MrLUs3yPYD6yJiNOAlcC5ks5s2ud9wK6IGAY+C1ydYTxmZtZCZokg6vYmX85NPpoXSF4P3JDcvg14uyRlFZOZmR0u06ohSXMkbQJ2AvdGxGNNu5wEbAOIiIPAy8Bgi8e5VF
JNUm1sbCzLkM3MKifTRBARr0XESmAJcIakNzft0urVf/NZAxFxbUSMRMTI0NBQFqGamVVWLn0EEbEbuB84t+mu7cBSAElHAccCL+URk5mZ1WVZNTQk6bjk9gLgHcAzTbvdB
bwnuf2HwH0RcdgZgZmZZSfLPoITgRskzaGecG6NiLslbQBqEXExcD1wk6RR6mcCF2YYj5mZtZBZIoiIzcBvtdh+VcPtV4Hzs4rBzMw681xDVnhe2McsW55iwgrNC/uYZc9n
BFZYXtjHLB9OBFZYXtjHLB9OBFZYXtjHLB9OBFZYXtjHLB9+s9gKzQv7mGXPicAKzwv7mGXLl4bMzCrOicDMrOKcCMzMKs6JwMys4pwIzMwqzonAzKzinAjMzCrOicDMrOK
cCMzMKs6JwMys4pwIzMwqzonAzKzinAjMzCrOicDMrOKcCMzMKs6JwMys4pwIOhjfu58nt+1mfO/+vngeM7NmXqFsGndu+glX3r6ZuQMDHJiY4JrzVrBu5UmlfR4zs1Z8Rt
DG+N79XHn7Zl49MMGe/Qd59cAEV9y+ueuv2PN6HjOzdpwI2ti+ax9zB6YOz9yBAbbv2lfK5zEza8eJoI0lxy/gwMTElG0HJiZYcvyCUj6PmVk7TgRtDC6cxzXnrWD+3AEWz
TuK+XMHuOa8FQwunFfK5zEza0cR0esYZmRkZCRqtVpuzze+dz/bd+1jyfELMv3nnNfzmFk1SdoYESOt7nPVUAeDC+fl8o85+cxM2uW2aUhSUslfV/S05KekvThFucvI+ll
SZuSj6uyisfMzFrL8ozgIPDfIuJYXh/btrvwYhYm2EcZmY2jczOCCLihYh4hYh4PLm9B3gacJeUmVnB5FI1JGk58FvAYy3uPkvSTyX9raTT2hx/saSapNrY2FiGkZqZVU/miUDSQuB24IqI+GXT3Y8DyyLiV4C/AXyn1EkAXoqIj7TZ503Ajo
gISWcAt1E/Q2gblKQx4Mcpw4MvpfnFFhC4R+zNZWzuzpphPVUUr/0QnRjwoAfHg+b7U14Yk1do1UBSR481W2eKF8
sXseLOVVbyZJYKIeAhQh32+AHwhqxjMzKwzzzVkZlZx/Z4Iru11ADPkeLNVtnihfDE73mxlEm/pJp0zM7Pu6vczAjMz68CJwMys4vo2EUh6TtKWZFbT/BYwSEnS/5a0U9IP
G7a9UdK9kp5NPh/fyxgbtYn3E5J+0jB77O/2MsZG7Wa/LeoYTxNvIcdY0nxJP0imh3lK0OvY4Vp4/2KpK0N47uy17E2kjRH0hOS7k6+zmR8+zYRJN4
WESsLWif8FeDcpm1/DnwvIk4Bvpd8XRRf4fB4AT6bjPHKpDu8KCZnv/8G4KzV/114EzgQ5J+g+KOcbt4oZhjvB9Yk0wPsxI4V9KZwNXU4z0F2AW8r4cxNmoXL8BHG8Z3U/uH6IkPU5
+wc1Im49vviaCwIuIB4KWmzeupd2OTfP79XIOaRpt4C2ua2W8LOcZlm6036vYmX85NPgJYQ32GACjW+LaLt7AkLQHeDVyXfC0yGt9+TgQB3CNpo6RLex1MSidExAtQ/8cA/
GKP40njMkmbk0tHhbjM0qxp9tvCj3GL2XoLOcbJZYtNwE7gXuD/Absj4mCyy3YKlWmya442IyfH9VDK+n5VUpGUCPwdcAUwkXw+S0fj2cyJYHRGnA++ifpp9dq8D6kNfBP4d
9VPtF4C/6m04h+sw+23hts3qvaGCEa9FxEpgCXAG8Outdss3qvaa45X0ZuBjwK8B/x54I3BlD0M8RNJaYGdEbGzc3GLXroxv3yaCiPhp8nkn8gE3qv6hFt0PSiQDJ5509jmd
aEbEj+eOaAL5Mwca4zey3hR3jVvEWfYwBImI3cD/19zaOkzQ5dc0S4Ke9iqudhnjPTS7JRUTsB/6G4ozvamCdpOeo23gn8cPrvKoS7gP
ckt98D3NnDWDqa/Iea+AMKNMbJ9dTrgacj4jMNdxVyjNvFW9QxljQk6bjk9gLgHdTf1/g+9ckkoVjj2yreZxpeFIj69fZCjG9EfCwilkTEcuBC4L6I+GMyGt++7CyW9CvUz
wKgPrHe1yLiUz0M7TCS/JmbBImI3cD/19zaOkzQ5dc0S4Ke9iqudhnjPTS7JRUTsB/6G4ozvamCdpOeAW6hfEvocGY1vXyYCSUervjwmko4G3gn8cPrvKoS7gP
dwv1yyxPABclr7Z7app47wOGqF922QS8v+FN5UKQdA5weUSszWp8+zIRmJlZen15acjMzNJzIjAzqzgnAjOzinMiMDOrOCcCM7OKcyIwm4ak15JZKX8o6RuSfqHNft+arFM
3KxuXj5pNQ9LeiFiY3P4qsLGp4vDGUvU/44m2j2GWdH5jMAsvQeBYUnLk3UD/hfwOLBU9fUvFgNIujiZxOxJSTcl24Yk3S7pn5KP1T08DrMpjuq8i5kl87u8C/h0sulXgT+JiA
8m90/u95vAX1Cf9PBFSW9M9vsc9Xn+/wAb2uRHKa7AgSfr4wFuOqdmhnkm+XnNmXxGPp+4lYe91XZg2c8GmPrm2o7SklxfhoBs1RnNhPhWdaRMSgzGMh0CHkOQrZ3OaAjs9o8eKrO5B34HRn/hbDGdjcGMsMlGkEjPZs3O
vuL1lMDDwCrImJfd8Mz03J+j8Csu74HXCBpEOqRyiCp5gSoPgEpN5Evjm9ypaGvjWrU5EZh1UUQ8BXwK+AdJ/TYW/M/D+XsVo1szlo2ZmFeczAjOzinMiMDOrOCcCM7OK
cyIwM6s4JwIzs4pzIjAzqzgnAjOzivv/Iee2ln4koHQAAAAASUVORK5CYII=\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "\n",
    "affordable_apps[reasonable].plot(kind='scatter', x='Price',y=\"Rating\")"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Let's Conclude that for reasonable apps there also isn't any significant relationship between price and rating."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 65,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "-0.11734743206298519"
      ]
     },
     "execution_count": 65,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps[reasonable].corr().loc['Rating','Price']"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "In the graph above, we see that there doesn't seem to be any clear relation between price and rating for the cheap apps. In
fact, the Pearson coefficient in this instance is around -0.1:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 66,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Find the mean price of the reasonable apps and assign it to cheap_mean.\n",
    "reasonable_mean = affordable_apps.loc[reasonable,\"Price\"].mean()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 67,
   "metadata": {},
   "outputs": [],
   "source": [
    "# Find the mean price of the reasonable apps and assign it to cheap_mean.\n",
    "\n",
    "affordable_apps.loc[reasonable,\"price_criterion\"] = affordable_apps[\"Price\"].apply(\n",
    "    lambda price: 1 if price < reasonable_mean else 0)"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Since affordable_apps has only around 700 rows and the genres column can take many different values, segmenting by this column
could spread our data too thin to extract any significant insights. Instead of simply ignoring it, we'll extract some information
from there and see where that leaves us.\n",
    "\n",
    "Looking at the possible values for this column, we see that ; isn't part of the name of any single genre:\n",
    "\n",
    "Let's create a column that counts the number of genres to which each app belongs. To do this, we'll use the Series.str.count()
string accessor. It takes a regular expression as input and it counts the number of occurences of the given pattern."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 68,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array(['Business', 'Communication', 'Dating', 'Education;Pretend Play',\n",
       "       'Education;Education', 'Entertainment', 'Food & Drink',\n",
       "       'Health & Fitness', 'Board', 'Puzzle', 'Strategy',\n",
       "       'Simulation;Education', 'Simulation', 'Action', 'Role Playing',\n",
       "       'Educational;Creativity', 'Educational;Pretend Play',\n",
       "       'Education;Creativity', 'Casual;Pretend Play', 'Casual;Education',\n",
       "       'Casual;Action & Adventure', 'Educational;Education',\n",
       "       'Board;Brain Games', 'Board;Pretend Play',\n",
       "       'Board;Action & Adventure', 'Education;Action & Adventure',\n",
       "       'Arcade;Action & Adventure', 'Card;Action & Adventure', 'Medical',\n",
       "       'Photography', 'Sports', 'Personalization', 'Productivity',\n",
       "       'Weather', 'Tools', 'Arcade', 'Racing', 'Education',\n",
       "       'Puzzle;Brain Games', 'Travel & Local', 'Lifestyle',\n",
       "       'Auto & Vehicles', 'Adventure', 'Card', 'Casual',\n",
       "       'News & Magazines', 'Shopping', 'Books & Reference', 'Social',\n",
       "       'Art & Design', 'Video Players & Editors', 'Finance',\n",
       "       'Adventure;Brain Games', 'Role Playing;Action & Adventure',\n",
       "       'Educational', 'Maps & Navigation', 'Parenting',\n",
       "       'Adventure;Action & Adventure', 'Casino', 'Education;Brain Games',\n",
       "       'Libraries & Demo', 'Action;Action & Adventure',\n",
       "       'Sports;Action & Adventure', 'Strategy;Action & Adventure',\n",
       "       'Racing;Action & Adventure', 'Books & Reference;Creativity',\n",
       "       'Music', 'Books & Reference;Education', 'Simulation;Pretend Play',\n",
       "       'Music;Music & Video', 'Role Playing;Education'], dtype=object)"
      ]
     },
     "execution_count": 68,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps['Genres'].unique()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 69,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0      1\n",
       "1      1\n",
       "2      1\n",
       "3      1\n",
       "4      1\n",
       "      ..\n",
       "793    1\n",
       "794    1\n",
       "795    1\n",
       "796    1\n",
       "797    1\n",
       "Name: genre_count, Length: 774, dtype: int64"
      ]
     },
     "execution_count": 69,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps[\"genre_count\"] = affordable_apps['Genres'].str.count(';')+1\n",
    "affordable_apps[\"genre_count\"]\n"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "We counted the occurrences of ; in the Genres column above for each row, which give us the number of separators, and we added
one to get the number of values."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 70,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th></th>\n",
       "      <th>Price</th>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>affordability</th>\n",
       "      <th>genre_count</th>\n",
       "      <th></th>\n",
       "    </tr>\n",
       "  </thead>\n",
       "  <tbody>\n",
       "    <tr>\n",
       "      <th rowspan=\"2\" valign=\"top\">cheap</th>\n",
       "      <th>1</th>\n",
       "      <td>2.519002</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>2</th>\n",
       "      <td>3.185833</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th rowspan=\"2\" valign=\"top\">reasonable</th>\n",
       "      <th>1</th>\n",
       "      <td>13.071911</td>\n",
       "    </tr>\n",
       "    <tr>\n",
       "      <th>2</th>\n",
       "      <td>6.865000</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "</div>"
      ],
      "text/plain": [
       "                               Price\n",
       "affordability genre_count           \n",
       "cheap         1             2.519002\n",
       "              2             3.185833\n",
       "reasonable    1            13.071911\n",
       "              2             6.865000"
      ]
     },
     "execution_count": 70,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Let's now see how the mean price varies across the number of genres\n",
    "genres_mean  = affordable_apps.groupby([\"affordability\",\"genre_count\"]).mean().loc[:,[\"Price\"]]\n",
    "genres_mean "
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Apps that belong to two genres are more expensive among the cheap apps and cheaper among the reasonable apps.\n",
    "\n",
    "For each segment, let's label the apps that cost less than their corresponding segments' mean with 1, and the others with 0:"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "For each segment in categories_mean, label the apps that cost less than its segment's mean with 1, and the others with 0.
Create a column called category_criterion in affordable_apps with these labels."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 71,
   "metadata": {},
   "outputs": [],
   "source": [
    "def label_genres(row):\n",
    "    \"\"\"For each segment in `genres_mean`,\n",
    "    labels the apps that cost less than its segment's mean with `1`\n",
    "    and the others with `0`.\"\"\"\n",
    "  \n",
    "    price = row['Price']\n",
    "    if price > genres_mean.loc[row['affordability'],row['genre_count']][0]:\n",
    "        return 1\n",
    "    else:\n",
    "        return 0\n",
    "\n",
    "affordable_apps[\"genre_criterion\"] = affordable_apps.apply(\n",
    "    label_genres, axis=\"columns\")\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 72,
   "metadata": {},
   "outputs": [],
   "source": [
    "categories_mean = affordable_apps.groupby([\"affordability\",'Category']).mean()[['Price']]\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 73,
   "metadata": {},
   "outputs": [],
   "source": [
    "def label_categories(row):\n",
    "    \"\"\"For each segment in `categories_mean`,\n",
    "    labels the apps that cost less than its segment's mean with `1`\n",
    "    and the others with `0`.\"\"\"\n",
    "    aff = row['affordability']\n",
    "    cat = row[\"Category\"]\n",
    "    price = row['Price']\n",
    "    if price > categories_mean.loc[(aff, cat)][0]:\n",
    "        return 1\n",
    "    else:\n",
    "        0\n",
    "        \n",
    "\n",
    "affordable_apps[\"category_criterion\"] = affordable_apps.apply(\n",
    "    label_categories, axis=\"columns\")"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# Technique of majority voting."
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "We can use a technique called majority voting, in which we decide whether an app's price should increase based on all
criteria."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 74,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "384    1.0\n",
       "123    0.0\n",
       "68     1.0\n",
       "152    0.0\n",
       "647    1.0\n",
       "Name: Result, dtype: float64"
      ]
     },
     "execution_count": 74,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "criteria = [\"price_criterion\", \"genre_criterion\", \"category_criterion\"]\n",
    "# to find mode per each column.\n",
    "affordable_apps[\"Result\"] = affordable_apps[criteria].mode(axis='columns').drop([1],axis=1)\n",
    "affordable_apps[\"Result\"].sample(5)"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "It seems to be working as intendend. Let's see how many apps are eligible for a price increase:"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 75,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "333.0\n",
      "0.43023255813953487\n"
     ]
    }
   ],
   "source": [
    "nr_eleigible = affordable_apps[\"Result\"].sum()\n",
    "print(nr_eleigible , nr_eleigible/affordable_apps.shape[0], sep = \"\\n\")"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "We see that 333 apps – roughly 43% of the paid apps – are eligible for a price increase. This is a significant number, and we
consider the possibility that increasing the price for these apps will have a significant impact."
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "We could potentially fall back on the number of installations as a proxy for this, but unfortunately, our Installs column just
gives us ranges and not exact numbers.\n",
    "\n",
    "In any case, given the lack of options, we'll fall back on this obviously faulty proxy in order to estimate the impact. In
addition to it not being the main goal of this prototype, the lack of data also makes it hard to decide what the optimal price for
each app is. Instead, we'll use the mean price of the affordability of the eligible apps when it is higher than the current
price.\n",
    "\n",
    "\n",
    "Recall that we've already computed the mean price for the cheap and reasonable apps, and they are stored respectively in
cheap_mean and reasonable_mean.\n",
    "\n",
    "    Create a column in affordable_apps called New Price that should be:\n",
    "        The maximum between Price and cheap_mean for the cheap apps, rounded to two decimal places\n",
    "        The maximum between Price and reasonable_mean for the reasonable apps, rounded to two decimal places\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 76,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0       4.99\n",
       "1       4.99\n",
       "2       4.99\n",
       "3       4.99\n",
       "4       3.99\n",
       "       ...  \n",
       "793     2.60\n",
       "794    12.92\n",
       "795    16.99\n",
       "796     2.60\n",
       "797     2.60\n",
       "Name: New price, Length: 774, dtype: float64"
      ]
     },
     "execution_count": 76,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "def new_price(row):\n",
    "    \n",
    "    if row[\"affordability\"] == \"cheap\":\n",
    "        return round(max(row[\"Price\"], cheap_mean), 2)\n",
    "    else:\n",
    "        return round(max(row[\"Price\"], reasonable_mean), 2)\n",
    "affordable_apps[\"New price\"] = affordable_apps.apply(new_price,axis=1)\n",
    "affordable_apps[\"New price\"] "
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Transform Installs into a numeric column:\n",
    "\n",
    "    Replace + and , with nothing.\n",
    "    Pass int to the Series.astype method to transform Installs in a numeric column.\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 77,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0      100000\n",
       "1      100000\n",
       "2      100000\n",
       "3      100000\n",
       "4      100000\n",
       "        ...  \n",
       "793       100\n",
       "794      1000\n",
       "795     10000\n",
       "796     10000\n",
       "797        50\n",
       "Name: Installs, Length: 774, dtype: int64"
      ]
     },
     "execution_count": 77,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps['Installs']= affordable_apps['Installs'].str.replace(\"[+,]\",\"\").astype(int)\n",
    "affordable_apps['Installs']"
   ]
  },
  {
   "cell_type": "raw",
   "metadata": {},
   "source": [
    "Compute the difference between the new price and the current price. Multiply it by Installs.\n",
    "\n",
    "Create a column in affordable_apps called Impact"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 78,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0          0.0\n",
       "1          0.0\n",
       "2          0.0\n",
       "3          0.0\n",
       "4          0.0\n",
       "        ...   \n",
       "793      161.0\n",
       "794     4930.0\n",
       "795        0.0\n",
       "796    14000.0\n",
       "797       78.0\n",
       "Name: Impact, Length: 774, dtype: float64"
      ]
     },
     "execution_count": 78,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "affordable_apps[\"Impact\"] = (affordable_apps[\"New price\"]-affordable_apps[\"Price\"])*affordable_apps[\"Installs\"]\n",
    "affordable_apps[\"Impact\"] "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 79,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "198582357.64999998"
      ]
     },
     "execution_count": 79,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "total_impact = affordable_apps[\"Impact\"].sum()\n",
    "total_impact "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.7.6"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```