



# Girls Who Code At Home

डेटा प्लेग्राउंड

Python में डेटा का जोड़-तोड़ (मेनिपुलेशन)

## गतिविधि अवलोकन

डेटा विज़ुअलाइज़ेशन, संख्या रूपी और शब्द/प्रतीक रूपी सूचनाओं की विशाल मात्रा से युक्त डेटा को चित्रात्मक कला में बदलने की प्रक्रिया है। ये कलाकृतियां डेटा के बारे में एक कहानी सुनाती हैं। विभिन्न निरूपण दर्शकों के सामने महत्वपूर्ण जानकारी पर प्रकाश/ज़ोर डाल सकते हैं। उदाहरण के लिए, [COVID-19 के मामलों](#) की संख्या खोजते समय, आप यह जानकारी कई रूपों में देख सकते हैं: लाइन ग्राफ, [क्लोरोपेथ मैप](#), टेबल, बार ग्राफ, और कई अन्य रूप।

इस गतिविधि में आप सीखेंगे कि Python का उपयोग करते हुए विभिन्न प्रकार के ग्राफ, जैसे लाइन, बार, पाई, हिस्टोग्राम, और स्कैटर प्लॉट से डेटा चित्रण कैसे करते हैं। हम विशिष्ट रूप से [Kickstarter](#) प्रोजेक्ट्स से संबंधित डेटा पर नज़र डालेंगे; किक्स्टार्टर एक क्राउडफंडिंग (बहुत से लोगों के योगदान से चलने वाला) प्लेटफॉर्म है। डेटा के साथ काम करते समय आजकल के डेटा साइंटिस्ट्स कई भूमिकाएं निभाते हैं। सबसे पहले तो उन्हें डेटा को क्लीन करना होता है, यानी अशुद्ध डेटा को हटाना होता है और डुप्लिकेट डेटा की तलाश करनी होती है, फिर उन्हें विभिन्न प्रकार के अल्गोरिथ्म का उपयोग करके डेटा का विश्लेषण करना होता है, और आखिर में उन्हें हितधारकों के सामने अपना डेटा प्रस्तुत करना होता है (डेटा पर निर्भर करते हुए, हितधारकों में नीति नियंता, व्यावसायिक लीडर, शोधकर्ता, डॉक्टर और जनसामान्य आदि लोग शामिल हो सकते हैं)।

यदि आप इस बारे में और जानना चाहते हैं कि [Kickstarter डेटा में जोड़-तोड़ \(मेनिपुलेशन\)](#) कैसे करें, तो हमारी निम्नलिखित गतिविधि देखें:

[डेटा प्लेग्राउंड](#)

[डेटा साइंस](#) का क्षेत्र बहुत बड़ा है, जो आर्टिफिशियल इंटेलिजेंस (कृत्रिम बुद्धिमत्ता), डेटा माइनिंग, बिग डेटा और मशीन लर्निंग की अवधारणाओं को आपस में जोड़ता है। ग्लासडोर ([Glassdoor](#)) के अनुसार, डेटा साइंटिस्ट्स की जॉब सबसे अधिक मांग वाली जॉब्स में से एक है जिनका औसत मूल वेतन \$110,000 होता है।

## सीखने के लक्ष्य

इस गतिविधि को पूरा कर लेने पर आप निम्नलिखित कर सकेंगे...

- ◆ कस्टम ग्राफ बनाने के लिए pandas में `plot()` विधि के विभिन्न मानदंड उपयोग में लाना।
- ◆ डेटा के प्रकार के आधार पर सर्वोत्तम ग्राफिकल (चित्रात्मक) निरूपण तय करना।
- ◆ यह समझना कि किसी डेटासेट में खोजबीन करने के लिए Kaggle और Jupyter Notebook का उपयोग कैसे करें।

## सामग्री

- ◆ [Kaggle Kickstarter Data](#)
- ◆ [डेटा चित्रण का सैंपल प्रोजेक्ट](#)
- ◆ **वैकल्पिक:** यदि आपने हमारी [डेटा प्लेग्राउंड](#) गतिविधि पूरी कर ली है, तो [डेटा प्लेग्राउंड सैंपल प्रोजेक्ट](#) देखकर पता करें कि आप अपने अंतिम परिणामों को किस प्रकार चित्रित कर सकते हैं।

## पूर्व ज्ञान

इस प्रोजेक्ट से शुरुआत करने से पहले, हमारी सलाह है कि आप:

- ◆ अपने शब्दों में यह स्पष्ट कर सकते हों कि चर राशि या [variable](#) क्या होता है और यह बता सकते हों कि किसी प्रोग्राम में उनका उपयोग कैसे किया जा सकता है।
- ◆ अपने शब्दों में यह स्पष्ट कर सकते हों कि सशर्त कथन या कंडीशनल स्टेटमेंट ([conditional statement](#)) क्या होता है और यह बता सकते हों कि किसी प्रोग्राम में उनका उपयोग कैसे किया जा सकता है।
- ◆ अपने शब्दों में यह स्पष्ट कर सकते हों कि विधि या मेथड/प्रकार्य या फंक्शन ([method/function](#)) क्या होते हैं और यह बता सकते हों कि किसी प्रोग्राम में उनका उपयोग कैसे किया जा सकता है।
- ◆ किसी टेक्स्ट आधारित भाषा, जैसे जावास्क्रिप्ट (JavaScript), Python, स्विफ्ट (Swift) आदि के उपयोग का अनुभव रखते हों।

यदि आपको Python के बारे में क्विक रिविज़र चाहिए हो तो हमारा सुझाव है कि आप हमारी यह गतिविधि देखें

[क्या मैं आपकी मदद कर सकती हूँ?](#)

## वुमन इन टेक स्पॉटलाइट: फ़रनेंदा वियेगस (Fernanda Viégas)



तस्वीर स्रोत: [Google](#)

डेटा/सूचनाओं को कला में कैसे बदलते हैं? जब आप सूचनाओं को प्रस्तुत करना चाहते हैं तो आप वेन आरेख, बार ग्राफ, टेबल आदि बनाते हैं या फिर चित्रों का उपयोग करके भी डेटा को निरूपित करते हैं। फ़रनेंदा वियेगस अपने काम में मशीन लर्निंग के उपयोग से डेटा को खूबसूरत विज़ुअलाइज़ेशन में बदलने के साथ प्रयोग करती हैं।

फ़रनेंदा ने अपनी यात्रा का आरंभ एक पारंपरिक ग्राफिक डिज़ाइनर के रूप में किया था। MIT मीडिया लैब से परिचय होने के बाद, वे कला को टेक्नॉलजी से जोड़ने वाले कलाकारों से मिलकर प्रेरित हुईं। उन्होंने कोडिंग सीखना शुरू किया और अंततः

**फ़्लोइंग मीडिया (Flowing Media)** नामक एक स्टार्टअप कंपनी की स्थापना की, जो डेटा विज़ुअलाइज़ेशन का उपयोग करके विचार व्यक्त करने और कहानियाँ सुनाने में विशेषज्ञता रखती है। यहीं से [Google](#) की उनके काम पर नज़र पड़ी और उन्हें वहाँ **PAIR** (पीपल एंड आर्टिफिशियल इंटेलिजेंस रिसर्च) टीम का नेतृत्व करने की जॉब मिल गई।

हर दिन दुनिया में लगभग 2.5 क्विंटिलियन बाइट डेटा उत्पन्न होता है और इसका मात्र 30-40% भाग शोध के लिए उपलब्ध होता है। यह तो बहुत सारा डेटा है! फ़रनेंदा **मशीन लर्निंग** और **आर्टिफिशियल इंटेलिजेंस (AI)** का उपयोग करके डेटा को क्रमबद्ध करती हैं और उससे जानकारी हासिल करती हैं, ताकि हमारे लिए उसे बेहतर ढंग से समझना आसान हो जाए। उसके बाद वे एक कला व डिज़ाइन फ्रेमवर्क का उपयोग करके डेटा को ऐसे रूपों में बदलती हैं जिनसे हमारे लिए उसे देख और समझ पाना आसान हो जाता है। Google में उनके काम का एक भाग है जावास्क्रिप्ट में **TensorFlow** लाइब्रेरी का विकास करना, जिससे दूसरों के लिए डेटा को स्वयं विज़ुअलाइज़ करने हेतु मशीन लर्निंग का उपयोग करना और आसान हो जाता है और उन्हें नए सिरे से कोड लिखने की ज़रूरत नहीं रहती है।

फ़रनेंदा का काम डेटा को देखने के हमारे तरीके को किस प्रकार प्रभावित करता है इस बारे में और जानने के लिए यह [वीडियो](#) देखें!

चीको के बारे में अधिक जानना चाहती हैं? [फ़रनेंदा की वेबसाइट](#) देखें जहाँ वे अपने कुछ कमाल के काय पर प्रकाश डालती हैं! बेहिचक यह [वीडियो](#) भी देखें जहाँ वे अपने बैकग्राउंड और Google में अपने काम के बारे में कुछ बातें बता रही हैं।

## झलक

एक कंप्यूटर वैज्ञानिक होना, कोडिंग में बेहतरीन होने की तुलना में अधिक है। इस बात के बारे में सोचने में थोड़ा समय बिताएं कि कैसे चीको और उनका काम उन शक्तियों से संबंधित है जिन पर महान कंप्यूटर वैज्ञानिक - बहादुरी, लचीलेपन, रचनात्मकता और उद्देश्य के निर्माण के दौरान ध्यान केंद्रित करते हैं।



### रचनात्मकता

फ़रनेंदा डेटा विज़ुअलाइज़ेशन के क्षेत्र में अपने करियर को गढ़ने के लिए ग्राफिक डिज़ाइन और कंप्यूटर साइंस के अपने दो जुनूनों को एक साथ लाती हैं। आपकी ऐसी किन दो या अधिक चीज़ों में रुचि है जिनका ऊपरी तौर पर एक-दूसरे से कोई संबंध दिखाई नहीं पड़ता है? क्या आप ऐसी गतिविधियाँ सोच सकते हैं जो आपकी रुचियों को आपस में जोड़ सकती हों?

परिवार के किसी सदस्य या मित्र के साथ अपनी प्रतिक्रियाएँ साझा करें। चर्चा में शामिल होने हेतु दूसरों को नैन्सी के बारे में अधिक पढ़ने के लिए प्रोत्साहित करें!

## चरण 1: बिग डेटा क्या होता है? (5-10 मिनट)

इंटरनेट द्वारा उत्पन्न लगभग 70% डेटा बेकार चला जाता है, पर क्यों? इस अनुभाग में चर्चा करेंगे कि **बिग डेटा** क्या होता है और हर उद्योग की कंपनियों के लिए यह मूल्यवान क्यों है।

यदि आप हमारी **डेटा प्लेग्राउंड** गतिविधि पहले ही पूरी कर चुके हैं, तो बेहिचक चरण 1 से 3 छोड़ दें।

### बिग डेटा (2 मिनट)

**बिग डेटा** का अर्थ है बिग डेटा, यानी बहुत सारा डेटा! हम बिग डेटा और बस सामान्य डेटा में भेद इसलिए करते हैं क्योंकि बिग डेटा के उत्पन्न होने की दर इतनी होती है कि उसका रखरखाव करना कठिन होता है। बिग डेटा कुछ टेराबाइट (TB) स्पेस घेर सकता है, जबकि सामान्य डेटा आमतौर पर दसियों या सैकड़ों गीगाबाइट (GB) स्पेस घेरता है। यानी सामान्य डेटा से लगभग 100 गुना बड़ा! प्रायः बिग डेटा को “साफ” करने या उपयोग और व्याख्या के लायक तैयार बनाने के लिए बहुत प्रयास करने पड़ते हैं। उदाहरण के लिए इस बारे में सोचें कि आप अपने मित्र को कोई साधारण सा वाक्यांश, जैसे कि “ok”, कैसे भेजेंगे। आप “okay”, “ok”, “k”, “kay”, या कुछ और लिखकर भेज सकते हैं! हमारे द्वारा अपरकेस (कैपिटल) या लोअरकेस (स्मॉल) अक्षरों और विशेष वर्ण के उपयोग के आधार पर भी इसके कई अन्य रूप हो सकते हैं। ये सभी संभावित विकल्प, डेटा की रचना करते हैं और हमें कंप्यूटर को यह पहचानने के लिए प्रशिक्षित करना होता है कि इन सभी शब्दों का अर्थ एक ही है।



डेटा की विशाल मात्रा उपलब्ध होने के बावजूद, उसका अधिकांश भाग पूर्वग्रहों (बायस) की मौजूदगी के कारण उपयोग योग्य नहीं माना जाता है। पूर्वग्रह (बायस) क्या होता है? डेटा में पूर्वग्रह तब उत्पन्न होता है जब कोई परिणाम विशेष, कुछ नतीजों के लिए अधिक अनुकूल होता है। ऐसा विभिन्न कारणों से हो सकता है, जिनमें से एक कारण यह हो सकता है कि एकत्र सूचनाएं, संपूर्ण जनसमूह का उचित प्रतिनिधित्व नहीं करती हैं। मान लें कि आप स्कूल में अपने लोगों का सर्वेक्षण कर रहे हैं। यदि आप खुद को सबसे पहले दिखने वाले मात्र 20 लोगों से प्रश्न पूछते हैं, तो क्या आपको लगता है कि आपका डेटा पूरे स्कूल का उचित प्रतिनिधित्व करता है? नहीं। संपूर्ण जनसमूह का उचित प्रतिनिधित्व करने के लिए न केवल यह महत्वपूर्ण है कि आप *किसका* या *किन चीजों* का सर्वेक्षण करते हैं, बल्कि यह भी महत्वपूर्ण है कि आप किस प्रकार की सूचनाएं एकत्र करते हैं। विविध डेटासेट प्राप्त करने के लिए, आपके जनसमूह में और, विश्लेषण करने वाली आपकी टीम में विविधता होना महत्वपूर्ण है। यदि आप डेटा पूर्वग्रहों के बारे में और जानना चाहते हैं तो एल्डर रिसर्च (Elder Research) का यह [लेख](#) या गूगल (Google) का यह [वीडियो](#) देखें।

- ◆ [डेटा साइंस में सांख्यिकी और संज्ञान संबंधी पूर्वग्रह: पूर्वग्रह \(बायस\) क्या होता है?](#) एल्डर रिसर्च (Elder Research) द्वारा
- ◆ [मशीन लर्निंग और मानवीय पूर्वग्रह वीडियो](#), Google द्वारा।
- ◆ [आर्टिफिशियल इंटेलिजेंस में पूर्वग्रहों से निपटना](#), The New York Times द्वारा



## चरण 1: बिग डेटा क्या होता है? (जारी)

### Kaggle के साथ शुरुआत (5-8 मिनट)



**Kaggle** एक वेबसाइट है जो एक ऑनलाइन समुदाय के योगदान से प्राप्त असली डेटा को होस्ट करती है। इस वेबसाइट पर होस्ट किए गए डेटा में [COVID-19](#), [YouTube वीडियो](#), [Google Play पर उपलब्ध एप्स](#), [स्तन कैंसर](#), [एवोकाडो की कीमतें](#), और [यहां तक कि पोकेमॉन \(Pokémon\)](#) तक के बारे में आंकड़े उपलब्ध हैं! यह वेबसाइट, आज हम जो निर्णय लेते हैं उन्हें प्रभावित करने वाले असली डेटा की खोजबीन के लिए एक अच्छी जगह है। Kaggle के अंदर, आप वेबसाइट में ही कोड लिखने के लिए किसी डेटासेट विशेष से संबंधित **नोटबुक** बना सकते हैं! यह अपने प्रोजेक्ट्स को व्यवस्थित करने का और असली लाइव प्रोजेक्ट्स में कार्य जमा करने का एक अच्छा तरीका है!

- **एक Kaggle अकाउंट बनाएं।** इस [लिंक](#) पर क्लिक करके Kaggle में एक अकाउंट बनाएं। या फिर, आप Kaggle वेबसाइट के ऊपरी दायें कोने में मौजूद **रजिस्टर (Register)** बटन पर क्लिक करके अकाउंट बना सकती हैं। *यदि आप 13 वर्ष से कम उम्र की हैं, तो आपको साइन अप करने के लिए अपने अभिभावक की अनुमति और ईमेल पते की आवश्यकता होगी।*



- **Kickstarter डेटासेट खोलें।** इस डेटासेट में सबसे ऊपर एक मेन हेडर होता है जिसमें डेटासेट का शीर्षक, रचयिता का नाम और अंतिम अपडेट का समय होता है। इसके ठीक नीचे कई विकल्प होते हैं: Data (डेटा), Tasks (कार्य), Notebooks (नोटबुक), Discussion (चर्चा), Activity (गतिविधि), और Metadata (मेटाडेटा)। Kaggle द्वारा हर डेटासेट के लिए प्रदत्त विशेषताओं के बारे में आप और जानकारी [यहां](#) पा सकते हैं।

## चरण 2: डेटासेट में खोजबीन करें (10-15 मिनट)

इससे पहले कि हम डेटा चित्रण के बारे में जानना शुरू करें, हमें समय निकाल कर उस डेटासेट को ठीक से जानना होगा जिसे हमने इस गतिविधि के लिए चुना है। यह समझना महत्वपूर्ण है कि किस डेटा को निरूपित किया जा रहा है और यह देखना भी कि कहीं कोई पूर्वग्रह तो मौजूद नहीं है।

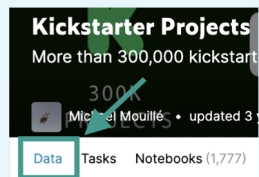
### Kickstarter प्रोजेक्ट्स (1 मिनट)



**Kickstarter** एक क्राउडफंडिंग (बहुत से लोगों के योगदान से चलने वाला) प्लेटफॉर्म है जहां व्यक्ति या छोटे-छोटे व्यवसाय सामुदायिक निवेश के जरिए रचनात्मक प्रोजेक्ट्स की फंडिंग कर सकते हैं। किसी प्रोजेक्ट में निवेश करना चाहने वालों को एक निश्चित राशि का संकल्प लेना होता है। उनके संकल्प के बदले में उन्हें उनके प्रोजेक्ट्स पूरे होने पर उपहार दिए जाते हैं। आप Kickstarter पर होस्ट किए गए प्रोजेक्ट्स के कुछ उदाहरण [यहां](#) देख सकती हैं।

आप शायद सोच रही होंगी, *कि हमने यह डेटासेट क्यों चुना?* Kickstarter डेटा में दो तरह के वैल्यूज़ (मान) होते हैं, पहला तो है **श्रेणीगत (कैटेगोरिकल)**, यानी ऐसा डेटा जिसे श्रेणियों में समूहबद्ध किया जा सकता है, और दूसरा है संख्यात्मक (न्यूमेरिकल)। हमने इसे चुना है क्योंकि यह आपको विभिन्न प्रकार के डेटा का एक उत्तम सारांश दिखाता है। इस गतिविधि में हम आपको उन *कुछ* चीज़ों के बुनियादी उदाहरण समझाएंगे जो आप दोनों प्रकार के वैल्यूज़ को विज़ुअलाइज़ करने यानी उन्हें दृष्टिगोचर बनाने के लिए कर सकते हैं।

### अपना दृश्य सेट अप करना (5-8 मिनट)



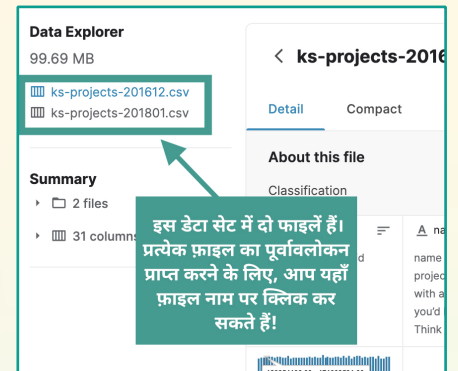
आइए, हमारे डेटा में गोता लगाएं! सबसे पहले, चलिए **Kaggle Kickstarter डेटासेट** को खोलते हैं। सुनिश्चित करें कि आप **Data** टैब पर हों। आपको ध्यान देना चाहिए कि सबसे ऊपर मौजूद हेडर बार पर “Data” शब्द **नीला** है और **अंडरलाइन** किया हुआ है। अब नीचे स्क्रॉल करते हुए **Data Explorer** वाले भाग में पहुंचें।

## चरण 2: डेटासेट में खोजबीन करें (जारी)

बायीं ओर नज़र डालें, आपको दिखेगा कि वहां दो फ़ाइल्स हैं, **ks-projects-201612.csv** और **ks-projects-201801.csv**.

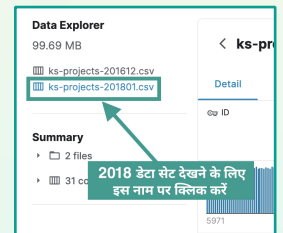
- ◆ **ks-projects-201612.csv**: इस डेटासेट में वे सारे Kickstarter प्रोजेक्ट्स हैं जो दिसंबर 2016 से पहले लॉन्च किए गए थे।
- ◆ **ks-projects-201801.csv**: इस डेटासेट में वे सारे Kickstarter प्रोजेक्ट्स हैं जो जनवरी 2018 से पहले लॉन्च किए गए थे।

चूंकि हम सबसे हाल के डेटा के साथ कार्य करना चाहते हैं, अतः हम केवल **ks-projects-201801.csv** के साथ कार्य करेंगे, संक्षेप में हम इसे 2018 डेटासेट कहेंगे।



चूंकि **ks-projects-201801.csv** में जनवरी 2018 से पहले के सारे प्रोजेक्ट्स हैं, इसलिए इस डेटासेट में **ks-projects-201612.csv** का सारा डेटा है और उससे भी अधिक डेटा है!

- ◆ **2018 Kickstarter डेटासेट खोलें।** Data Explorer में, बायीं ओर मौजूद **ks-projects-201801.csv** नाम पर क्लिक करें। आपको यह नाम नीले रंग में हाइलाइट किया हुआ दिखना चाहिए।
- ◆ **दृश्य को विस्तार दें।** डेटा एक्सप्लोरर विंडो के ऊपरी दायें कोने में मौजूद बॉक्स आइकन पर क्लिक करें।
- ◆ **सारांश सूचना छिपाएं।** डेटासेट के नाम के बायीं ओर मौजूद तीर के निशान पर क्लिक करें। इससे बायीं तरफ बंद हो जानी चाहिए, जिससे हमें डेटा को देखने के लिए और अधिक स्थान मिल जाएगा!



इस दृश्य से हमें डेटा का और अंदर मौजूद कुछ वैल्यूज़ का एक त्वरित सारांश दिखता है। ध्यान दें कि इसमें पूरा डेटासेट नहीं दिखता है क्योंकि वह बहुत विशाल होता है! असल में, इस पूरे डेटा सेट में 375,765 Kickstarter प्रोजेक्ट्स की सूचनाएं हैं!

### Kickstarter प्रोजेक्ट फ़ीचर्स (1 मिनट)

इस डेटासेट के कॉलम्स (स्तंभों) को डेटा के फ़ीचर्स (विशेषताएं) कहा जाता है। फ़ीचर्स से हमें पता चलता है कि प्रत्येक प्रोजेक्ट के लिए किस प्रकार की सूचना दर्ज हुई है। डेटासेट की प्रत्येक रॉ (पंक्ति) एक अकेले Kickstarter प्रोजेक्ट को, या एक **एंटिटी को दर्शाती है**। उदाहरण के लिए, यदि हमारे पास मनुष्यों का कोई डेटासेट होता, तो उस डेटासेट में व्यक्ति एक **एंटिटी** होता और इस डेटासेट में हम जो फ़ीचर्स शामिल करते थे कुछ यूं होते: व्यक्ति का नाम, आंखों का रंग, बालों का रंग, जन्म दिनांक, आदि।

उदाहरण

Name	Eye Color	Hair Color	Date of Birth
Reshma Saujani	Brown	Brown	November 18, 1975

एक अकेली **एंटिटी**, रेशमा सौजानी का और उसे परिभाषित करने वाले कुछ **फ़ीचर्स** का उदाहरण।

## चरण 2: डेटासेट में खोजबीन करें (जारी)

आइए इस Kickstarter डेटा में **फ़ीचर्स**, यानी कॉलम्स को एक-एक करके देखते हैं! इस डेटा सेट में कुल 15 फ़ीचर्स हैं: **ID, name, category, main\_category, currency, deadline, goal, launched, pledged, state, backers, country, usd pledged, usd\_pledged\_real, usd\_goal\_real**. पहली रो (पंक्ति) हमें फ़ीचर का नाम और उसका संक्षिप्त विवरण बताती है, वहीं दूसरी रो में प्रत्येक फ़ीचर की कुछ वैल्यूज़ का एक सारांश दृश्य है। एक मिनट निकाल कर प्रत्येक फ़ीचर और उसके संक्षिप्त विवरण पर ग़ौर करें।

**ध्यान दें:** डेटा एक्सप्लोरर में आपको 15 में से केवल 10 कॉलम दिखेंगे, डेटासेट के ऊपरी दायें कोने में एक ड्रॉप-डाउन मेन्यू है जिससे आप अतिरिक्त कॉलम देख सकते हैं। आपको वे अतिरिक्त 5 कॉलम ढूंढने के लिए मेन्यू में नीचे स्क्रॉल करना होगा जो डिफ़ॉल्ट दृश्य में प्रदर्शित नहीं हो रहे हैं।



## चरण 3: Python और Pandas का परिचय (20-25 मिनट)

डेटा की प्रोसेसिंग और विश्लेषण के लिए बहुत से टूल्स और भाषाओं का उपयोग करते हैं, जैसे **R**, **स्ट्रक्चर्ड क्वेरी लैंग्वेज (या SQL)**, और **Python**। इस गतिविधि में हम **Pandas** नामक एक Python लाइब्रेरी की मदद से हमारे डेटा में जोड़-तोड़ यानी मेनिपुलेशन करने के लिए Python प्रोग्रामिंग भाषा के उपयोग पर ध्यान केंद्रित करेंगे।

### Python (2 मिनट)



**Python** एक टेक्स्ट-आधारित प्रोग्रामिंग भाषा है, यानी हमें सारे कमांड टाइप करने होंगे! कई प्रोग्रामर्स Python का उपयोग इसलिए चुनते हैं क्योंकि इसे सीखना और समझना आसान है। Python एक **मुक्त स्रोत (ओपन सोर्स)** भाषा है, यानी यह सभी के द्वारा उपयोग और आवश्यकतानुसार संशोधन के लिए मुक्त रूप से उपलब्ध है। अपडेट्स कैसे स्वीकरी जाती हैं और भाषा पर लागू की जाती हैं इस बारे में कड़े दिशानिर्देश मौजूद हैं, पर मूल रूप से कहीं तो कोई भी इस भाषा के विकास में योगदान दे सकता है!

चूंकि Python एक ओपन सोर्स भाषा है, अतः इस कारण से प्रोग्रामर्स के समुदाय ने कई अतिरिक्त लाइब्रेरीज़ विकसित कर ली हैं। **लाइब्रेरी** का अर्थ **मेथड्स** (विधियों) और **वेरिएबल्स** (चर राशियों) के एक संकलन से होता है। लाइब्रेरी कोड लिखना आसान बना देती है, क्योंकि हम किसी काम को करने के लिए कई पंक्तियों का कोड लिखने की बजाय उसी काम के लिए लाइब्रेरी में मौजूद कमांड्स का उपयोग कर सकते हैं। इस गतिविधि में हम pandas लाइब्रेरी का उपयोग करेंगे। इस विशेष लाइब्रेरी को विशेष रूप से डेटा साइंटिस्ट्स के लिए बनाया गया है जिससे वे सर्च, फ़िल्टर, तुलना, संशोधन, और डेटासेट से सूचना हटाने जैसे साधारण कार्य करने के लिए ढेर सारी पंक्तियों का कोड लिखने के झंझट से मुक्त होकर डेटासेट का आसानी से विश्लेषण कर सकते हैं। इससे पहले कि हम यह जानें कि विशिष्ट रूप से हमारे डेटा पर क्रियाएं करने के लिए हम pandas का उपयोग कैसे करेंगे, आइए Kaggle में एक नई नोटबुक बनाकर अपने प्रोग्रामिंग परिवेश को व्यवस्थित कर लेते हैं।

### चरण 3: Python और Pandas का परिचय (जारी)

#### नई नोटबुक बनाना (5-8 मिनट)

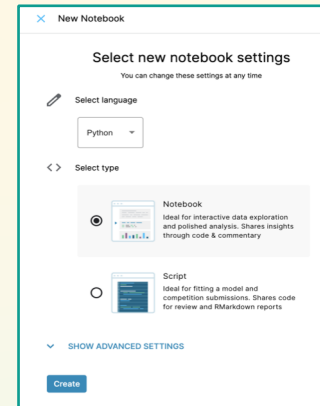
चलिए वापस हमारे [Kickstarter डेटासेट](#) पर लौटते हैं।

- **नई नोटबुक बनाएं।** दायीं ओर मेन हेडर इमेज के नीचे मौजूद **New Notebook** बटन पर क्लिक करें। ऐसा करने पर आपके सामने नई नोटबुक की सेटिंग्स चुनने की स्क्रीन खुलेगी। सुनिश्चित करें कि आपकी नोटबुक में निम्नलिखित हो:

- ◆ **भाषा:** Python
- ◆ **प्रकार:** नोटबुक

सेटिंग्स की पुष्टि करें और **Create** पर क्लिक करें।

हो सकता है कि आपका डेटासेट अभी-भी पूर्ण-दृश्य में प्रदर्शित हो रहा हो। इसे **मिनिमाइज़** करने के लिए, ऊपरी दायें कोने में मौजूद **बॉक्स आइकन** पर क्लिक करें।



यदि आप पहले Python में प्रोग्रामिंग कर चुके हैं, तो आपको यह “नोटबुक”, Trinket या अन्य Python एडिटर्स में प्रोग्रामिंग करने से अलग दिख सकती है। Kaggle, Python में प्रोग्रामिंग के लिए [Jupyter Notebook](#) नामक टूल का उपयोग करता है। Jupyter Notebook एक टूल/ऐप्लिकेशन है जिसका उपयोग कोड, टेक्स्ट, और विजुअलाइज़ेशन, सब कुछ एक साथ एक जगह दिखाने के लिए किया जा सकता है। पूरे प्रोग्राम को चलाने की विवशता के बिना कोड के ब्लॉक्स को चलाना आसान होता है।

- **अपनी नोटबुक को नया नाम दें।** यह करने के लिए, अपनी स्क्रीन के ऊपरी बायें भाग में मौजूद डिफ़ॉल्ट नाम पर क्लिक करें, और मौजूद टेक्स्ट को अपने नए शीर्षक से बदल दें। सुनिश्चित करें कि आपका नाम हमें इस बारे में कुछ बताए कि इस प्रोजेक्ट में आप क्या संपन्न करने जा रहे हैं, जैसे, “डेटा की छानबीन को चित्रित करना”। आप चाहें तो शीर्षक में अपना नाम भी शामिल कर सकते हैं, सुनिश्चित करें कि आप या तो केवल प्रथमाक्षर दिखाएं या अपना पहला नाम और अंतिम नाम का प्रथमाक्षर दिखाएं।

#### स्टार्टर कोड को जानें (3-5 मिनट)

आइए थोड़ा समय निकाल कर आपकी नोटबुक में शामिल किए गए स्टार्टर कोड के कुछ हिस्से पर नज़र डालते हैं। आपको पूरे स्टार्टर कोड में कुछ कोड की पंक्तियाँ ऐसी दिखेंगी जो **#** सिम्बल से शुरू होती हैं। इन कोड की पंक्तियों को **कोड कमेंट्स** कहा जाता है। प्रोग्रामर्स इनका उपयोग अपने कोड को व्यवस्थित करने और उसे पढ़ने में आसान बनाने के लिए करते हैं। Python में, **#** सिम्बल के बाद लिखा समस्त टेक्स्ट, कोड कमेंट माना जाता है और फिर उसे नील-हरित रंग में रंग दिया जाता है।

चलिए अब कोड की शुरुआती पंक्तियों पर नज़र डालते हैं, जो **import** कीवर्ड से शुरू होती हैं।

PYTHON	विवरण
<pre>import numpy as np import pandas as pd</pre>	<ul style="list-style-type: none"><li>◆ <b>import</b>: यह कीवर्ड कंप्यूटर को बताता है कि हम एक Python लाइब्रेरी का उपयोग कर रहे हैं।</li><li>◆ <b>as</b>: यह कीवर्ड, पैकेज को एक उपनाम देता है। यह एक वैकल्पिक चरण है पर इससे प्रोग्रामिंग आसान हो सकती है।</li><li>◆ <b>numpy/np</b>: इस Python लाइब्रेरी का उपयोग डेटासेट पर गणितीय संक्रियाएं करने के लिए होता है। हमने इस पैकेज को <b>np</b> उपनाम दिया है।</li><li>◆ <b>pandas/pd</b>: इस Python लाइब्रेरी का उपयोग डेटासेट को विश्लेषण हेतु एक अधिक आसान उपयोग वाले फ़ॉर्मेट में बदलने के लिए होता है। हमने इस पैकेज को <b>pd</b> उपनाम दिया है।</li></ul>

हम इस गतिविधि में **numpy** लाइब्रेरी का उपयोग नहीं करेंगे। जब आप डेटा एनालिटिक्स को स्वयं ही जानना-समझना जारी रखें, तो इस बारे में बेहिचक और जानें कि डेटा साइंटिस्ट numpy लाइब्रेरी का उपयोग कैसे करते हैं।



### चरण 3: Python और Pandas का परिचय (जारी)

और आखिर में, चलिए अंतिम कोड की पंक्तियाँ देखते हैं।

PYTHON	विवरण
<pre>import os for dirname, _, filenames in os.walk('/kaggle/input'):     for filename in filenames:         print(os.path.join(dirname, filename))</pre>	Kaggle में नोटबुक की एक सबसे अच्छी विशेषता यह है कि आप अतिरिक्त चरणों को पूरा करने की विवशता के बिना डेटासेट का उपयोग आसानी से कर सकते हैं। कैसे? आपकी जिस डेटासेट में रुचि है उसे Kaggle आपकी नोटबुक के साथ पहले ही संबद्ध कर देता है! यह वैकल्पिक कोड की पंक्तियाँ हमें यह जानने में मदद करती हैं कि हमारे डेटासेट के लिए फ़ाइलनेम क्या है।

#### कोड चलाना (2 मिनट)

सबसे पहले, कोड ब्लॉक के अंदर कहीं भी क्लिक करें। विंडो के बायीं ओर दिखने वाले **नीले प्ले बटन** ► पर क्लिक करें। इससे कोड ब्लॉक के अंदर मौजूद सभी कोड की पंक्तियाँ चलनी (रन होनी) चाहिए और विंडो के ठीक नीचे आउटपुट (यदि कोई हो तो) दिखना चाहिए। इन कोड की पंक्तियों को चलाने पर आपको यह आउटपुट मिलना चाहिए:

```
/kaggle/input/kickstarter-projects/ks-projects-201801.csv
/kaggle/input/kickstarter-projects/ks-projects-201612.csv
```

ये हमारे डेटासेट की फ़ाइल्स के नाम हैं! अब हम स्टार्टर कोड में जो कुछ शामिल है उसके बारे में और हमारी नोटबुक में कोड चलाने के तरीके के बारे में थोड़ा समझ चुके हैं, तो आइए, कोडिंग शुरू करते हैं!

#### Kickstarter डेटासेट इम्पोर्ट करना (10-15 मिनट)

अब हम हमारे डेटासेट की फ़ाइल्स के नाम जान चुके हैं, और अब हमें डेटा को हमारे प्रोग्राम में वास्तव में इम्पोर्ट करना यानी लाना होगा। इस समय यह एक अलग फ़ाइल में मौजूद है, पर हमें इस जानकारी को हमारे Python प्रोग्राम से जोड़ना होगा। इसे करने के लिए हम **pandas** लाइब्रेरी में मौजूद **read\_csv()** विधि का उपयोग करेंगे। याद रखें कि **मैथड/फंक्शन** निर्देशों (कोड की पंक्तियाँ) का एक सेट होता है जो एक विशिष्ट कार्य करता है। आइए इस मैथड के सिन्टेक्स (वाक्य-रचना) को छोटे-छोटे टुकड़ों में समझें।

**CSV**, यानी कॉमा सेपरेटेड वैल्यूज़, एक प्रकार की फ़ाइल होती है जिसमें डेटा, साधारण टेक्स्ट के रूप में रखा जाता है। **pandas** लाइब्रेरी, CSV फ़ाइल्स को आसानी से पढ़ सकती है और उन्हें टेबल जैसे फ़ॉर्मेट में बदल सकती है जिसे हम आसानी से पढ़ सकते हैं और उसमें जोड़-तोड़ कर सकते हैं।

PYTHON	विवरण
<pre>pd.read_csv("filename")</pre>	<ul style="list-style-type: none"><li>◆ <b>pd</b>: यह कीवर्ड कंप्यूटर को बताता है कि हम <b>pandas</b> लाइब्रेरी में मौजूद एक मैथड का उपयोग कर रहे हैं। हम <b>pandas</b> की बजाय <b>pd</b> का उपयोग करते हैं क्योंकि हमने हमारे प्रोग्राम की शुरुआत में लाइब्रेरी इम्पोर्ट करते समय लाइब्रेरी को यही उपनाम दिया था।</li><li>◆ <b>.</b>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मैथड का उपयोग कर रहे हैं।</li><li>◆ <b>read_csv()</b>: <b>pandas</b> में मौजूद यह मैथड, CSV फ़ाइल को पढ़ता है और उसे टेबल जैसे फ़ॉर्मेट में बदल देता है, ताकि Python में उसका उपयोग आसान हो जाए।</li><li>◆ <b>"filename"</b>: हमें कंप्यूटर को बताना होगा कि कौनसी फ़ाइल खोलनी है! यहां CSV फ़ाइल का फ़ाइलनेम लिखें। हम नाम कोटेशन मार्क्स (सिंगल या डबल कोट्स) में लिखते हैं क्योंकि यह एक नाम है।</li></ul>



### चरण 3: Python और Pandas का परिचय (जारी)

- **एक नया कोड ब्लॉक जोड़ें।** हमारी समस्या को छोटे-छोटे भागों में तोड़ते समय हमने जो उप-समस्याएं तय की थीं उनके अनुसार अपने कोड को व्यवस्थित करने में कोड ब्लॉक्स बहुत उपयोगी होते हैं। हो सकता है कि आपकी नोटबुक में एक खाली कोड ब्लॉक पहले से हो, जिसे एक लाइट-ग्रे बॉक्स से दिखाया जाता है और बॉक्स के बायीं तरफ `[]` सिम्बल होता है।

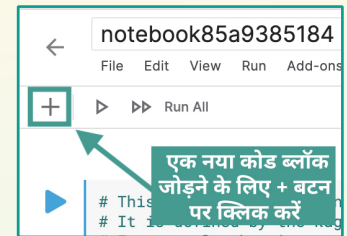
नया कोड ब्लॉक दो प्रकार से जोड़ सकते हैं:

- ◆ नोटबुक के ऊपरी मेन्यू में **+** बटन पर क्लिक करें।
- ◆ अपना माउस कर्सर अपने अंतिम कोड ब्लॉक के नीचे ले जाएं। आपको एक विकल्प दिखेगा जिसमें **+ Code** लिखा होगा। नीचे नया कोड ब्लॉक बनाने के लिए यह बटन चुनें।

- **Kickstarter डेटासेट इम्पोर्ट करें।** अब हमने नया कोड ब्लॉक बना लिया है, और अब समय है `read_csv()` मेथड का उपयोग करके हमारे डेटासेट को इम्पोर्ट करने का। पर ज़रा रुकें, हमें हमारे डेटासेट का फ़ाइल नेम चाहिए। याद रखें कि पहला ब्लॉक चलाने पर, उसके आउटपुट में हमारे डेटासेट के फ़ाइल नेम्स आते हैं। असल में, इसने आउटपुट में दो नाम दिए, 2016 और 2018 डेटासेट। हम केवल 2018 Kickstarter डेटासेट का उपयोग करेंगे क्योंकि यही सबसे हालिया जानकारी है। अपने नए कोड ब्लॉक में, `read_csv()` मेथड का उपयोग करके 2018 डेटासेट को इम्पोर्ट करें। आपके पिछले कोड ब्लॉक के आउटपुट से 2018 डेटासेट का फ़ाइल नेम **कॉपी और पेस्ट** करें।

```
pd.read_csv("/kaggle/input/kickstarter-projects/ks-projects-201801.csv")
```

- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपको अपने डेटासेट का एक छोटा सा स्नैपशॉट दिखना चाहिए (उसके जैसा जो आपने डेटा एक्सप्लोरर में देखा था)।



#### परिणाम

ID	name	category	main_category	currency
0	1000002330 The Songs of Aesha & Abdullah	Poetry	Publishing	GBP
1	1000003930 Greeting From Earth: ZCAC Arts Capsule For ET	Narrative Film	Film & Video	USD
2	1000004038 Where is Hank?	Narrative Film	Film & Video	USD
3	1000007540 ToshiCapital Records Needs Help to Complete Album	Music	Music	USD
4	100011046 Community Film Project: The Art of Neighborhood...	Film & Video	Film & Video	USD
...	...	...	...	...
378656	999976400 ChknTruk Nationwide Charity Drive 2014 (Canceled)	Documentary	Film & Video	USD
378657	999977640 The Tribe	Narrative Film	Film & Video	USD
378658	999986353 Walls of Remedy: New Lesbian Romantic Comedy I...	Narrative Film	Film & Video	USD
378659	999987933 BioDefense Education Kit	Technology	Technology	USD
378660	999988282 Nou Renmen Ayiti! We Love Haiti!	Performance Art	Art	USD

378661 rows x 5 columns

#### डीबर्गिंग के सुझाव

- ◆ जांचें कि आपने सही 2018 फ़ाइल नेम शामिल किया हो: `/kaggle/input/kickstarter-projects/ks-projects-201801.csv`
- ◆ सुनिश्चित करें कि फ़ाइल नेम, कोटेशन मार्क्स के अंदर हो
- ◆ याद रखें कि Python में, स्पेलिंग मायने रखती है! जांचें कि न केवल हर शब्द की स्पेलिंग सही हो, बल्कि यह भी कि वह सही केस (अपरकेस या लोअरकेस) में हो।
- ◆ जांचें कि pandas के मेथड को कॉल करते समय आपने पीरियड (फुल स्टॉप) शामिल किया हो।
- ◆ जांचें कि आपने कोई अतिरिक्त कोष्ठक ( ) न टाइप कर दिए हों। आप देखेंगे कि जब आप ( सिम्बल टाइप करते हैं तो नोटबुक अपने-आप कोलॉजिंग ब्रैकेट ) जोड़ देती है। इसलिए हो सकता है कि आप गलती से अतिरिक्त ब्रैकेट टाइप कर जाएं।

### चरण 3: Python और Pandas का परिचय (जारी)

- अपना डेटासेट `ds` नामक चर राशि में स्टोर करें। हमारा काम पूरा होने को है! हमने हमारे डेटासेट को हमारे Python प्रोग्राम से लिंक कर दिया है, अब हमें इसे एक चर राशि में स्टोर करना है। याद रखें कि कंप्यूटर प्रोग्राम में सूचनाओं (डेटा) को स्टोर करने के लिए चर राशि का उपयोग होता है। अपना डेटासेट `ds` नामक चर राशि में स्टोर करें (जो डेटासेट का संक्षिप्तीकरण है)।

```
ds = pd.read_csv("/kaggle/input/kickstarter-projects/ks-projects-201801.csv")
```

- `info()` मेथड का उपयोग करके अपने डेटासेट से संबंधित जानकारी प्रिंट करें। हम इस डेटासेट के फ़ीचर्स और रो (पंक्तियों) की संख्या का एक त्वरित सारांश पाने के लिए `info()` मेथड का उपयोग कर सकते हैं।

PYTHON	विवरण
<code>ds.info()</code>	<ul style="list-style-type: none"> <li>◆ <code>ds</code>: हम <code>info()</code> मेथड का उपयोग हमारे चर राशि <code>ds</code> पर करते हैं, न कि पूरी pandas लाइब्रेरी पर। ऐसा इसलिए है क्योंकि हम हमारे विशिष्ट डेटासेट के बारे में जानकारी पाना चाहते हैं।</li> <li>◆ <code>info()</code>: pandas में मौजूद यह मेथड, डेटासेट की जानकारी लेकर आती है, जिसमें फ़ीचर्स, रो (पंक्तियाँ), और डेटा टाइप शामिल हैं।</li> </ul>

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। इससे आपके डेटासेट के बारे में कुछ जानकारी आउटपुट में आनी चाहिए, जिसमें एंट्रीज़ की संख्या (या Kickstarter प्रोजेक्ट्स की संख्या) और फ़ीचर्स की संख्या शामिल है। इसमें प्रत्येक फ़ीचर की वैल्यूज़ का टाइप (संख्या या शब्द) भी शामिल है और उन एंट्रीज़ की संख्या भी जो “नॉन-नल (non-null)” हैं यानि खाली नहीं हैं।

#### परिणाम

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 378661 entries, 0 to 378660
Data columns (total 15 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    378661 non-null  int64
1   name                  378657 non-null  object
2   category              378661 non-null  object
3   main_category         378661 non-null  object
4   currency              378661 non-null  object
5   deadline              378661 non-null  object
6   goal                  378661 non-null  float64
7   launched              378661 non-null  object
8   pledged               378661 non-null  float64
9   state                 378661 non-null  object
10  backers               378661 non-null  int64
11  country               378661 non-null  object
12  usd pledged           374864 non-null  float64
13  usd_pledged_real      378661 non-null  float64
14  usd_goal_real         378661 non-null  float64
dtypes: float64(5), int64(2), object(8)
memory usage: 43.3+ MB
```

#### डीबगिंग के सुझाव

- ◆ Name 'ds' is not defined: आपकी नोटबुक का हर कोड ब्लॉक अलग-अलग चलता है, कभी-कभी हो सकता है कि आपकी नोटबुक प्रोग्राम में अपना स्थान भूल जाए और आपके लिए आपके **सभी कोड ब्लॉक्स को चलाना** आवश्यक कर दे। यह करने के लिए, अपनी नोटबुक में सबसे ऊपर दिए गए दो तीरों वाले बटन ▶ को क्लिक करके सारे कोड ब्लॉक्स चलाएं।
- ◆ याद रखें कि Python में, स्पेलिंग मायने रखती है! जांचें कि न केवल हर शब्द की स्पेलिंग सही हो, बल्कि यह भी कि वह सही केस (अपरकेस या लोअरकेस) में हो।
- ◆ जांचें कि किसी भी मेथड को कॉल करते समय आपने पीरियड (फ़ुल स्टॉप) शामिल किया हो।

## चरण 4: Pandas में डेटा को समझना (15-20 मिनट)

विजुअल ग्राफ की कोडिंग कैसे करते हैं यह सीखना शुरू करने से पहले, आइए हम थोड़ा समय निकाल कर विभिन्न प्रकार के डेटा को जानते हैं और यह जानते हैं कि pandas लाइब्रेरी किसी डेटासेट की प्रोसेसिंग कैसे करती है।



### श्रेणीगत (कैटेगोरिकल) बनाम संख्यात्मक (न्यूमेरिकल) डेटा (2-4 मिनट)

डेटा को दो भिन्न प्रकारों में वर्गीकृत किया जा सकता है: **श्रेणीगत (कैटेगोरिकल)** या **संख्यात्मक (न्यूमेरिकल)**। **श्रेणीगत डेटा** ऐसा डेटा होता है जिसे श्रेणियों या समूहों में बांटा जा सकता है, वहीं **संख्यात्मक डेटा** ऐसा डेटा होता है जिसे संख्या रूप में व्यक्त किया जा सकता है। संख्यात्मक डेटा में ऐसा डेटा होना चाहिए जो किसी निश्चित रेंज के अंदर के सतत मान (कंटीन्युअस वैल्यू) को निरूपित करता हो (यानि कोई भी संख्या छोड़ी न जाए)। कभी-कभी संख्याएं, श्रेणियों का प्रतिनिधित्व कर सकती हैं, उदाहरण के लिए, किसी सर्वेक्षण में भाग लेने पर आपसे अपने अनुभव को 1 से 5 के बीच की एक संख्या से रेटिंग देने को कहा जाता है। यहां हम देख सकते हैं कि अनुभव को 1 से 5 की संख्याओं द्वारा *समूहबद्ध* किया गया है। थोड़ा रुकें और निम्नलिखित उदाहरणों को श्रेणीगत (कैटेगोरिकल) या संख्यात्मक (न्यूमेरिकल) में वर्गीकृत करने की कोशिश करें।

- ◆ बिल्लियों की लंबाइयां, सेंटीमीटर में
- ◆ एक कक्षा के अंतिम ग्रेड (A, B, C, D, या F)
- ◆ वर्ष के महीनों को संख्याओं (1-12) द्वारा निरूपित करना
- ◆ नींद की अवधि, घंटों में
- ◆ लोगों के पसंदीदा रंग

नीचे समाधान देखने से पहले यहां थोड़ा रुकें।

- ◆ बिल्लियों की लंबाइयां, सेंटीमीटर में → **संख्यात्मक**
- ◆ एक कक्षा के अंतिम ग्रेड (A, B, C, D, या F) → **श्रेणीगत**
- ◆ वर्ष के महीनों को संख्याओं (1-12) द्वारा निरूपित करना → **श्रेणीगत**
- ◆ नींद की अवधि, घंटों में → **संख्यात्मक**
- ◆ लोगों के पसंदीदा रंग → **श्रेणीगत**

### Pandas DataFrame (5-8 मिनट)

pandas में, डेटा को एक टेबल जैसे ऑब्जेक्ट में स्टोर किया जाता है जिसे **DataFrame** कहते हैं। यह डेटा को उसी तरह स्टोर करता है जैसा हमने Kaggle पर डेटा एक्सप्लोरर में देखा था। DataFrame, डेटा की संरचना में मौजूद रो (पंक्तियों) और कॉलम (स्तंभों) की नकल करता है। डेटा को एक्सेस करने के लिए, हम **[]** सिम्बल्स का उपयोग करते हैं। यदि आप जावास्क्रिप्ट में **Arrays** से और Python में **Lists** से परिचित हैं, तो यह भी लगभग वैसी ही अवधारणा है।

	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

चूंकि हमारे डेटासेट में कुल **15 फ्रीचर्स**, या कॉलम्स हैं, तो हो सकता है कि हम डेटासेट के केवल कुछ ही फ्रीचर्स पर फोकस करना चाहें। आइए अभ्यास करें कि हम विशिष्ट फ्रीचर्स चुनने के लिए **[]** सिम्बल्स का किस प्रकार उपयोग कर सकते हैं।

तस्वीर स्रोत: [GeeksforGeeks](https://www.geeksforgeeks.org/)

## चरण 4: Pandas में डेटा को समझना (जारी)

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें। या तो अपना माउस कर्सर अपने अंतिम कोड ब्लॉक के नीचे ले जाएं और +Code बटन दबा दें, या फिर अपनी नोटबुक के ऊपरी बायें भाग में मौजूद + बटन दबा दें।
- [] सिम्बल्स का उपयोग हमारे डेटासेट चर राशि ds पर करके 'state' फ़ीचर को चुनें। याद रखें कि हमने हमारे डेटासेट के रेफ़रेंस को ds नामक चर राशि में स्टोर किया है। यहां हम [] सिम्बल्स का उपयोग यह दिखाने के लिए करते हैं कि हम डेटासेट से सूचनाएं चुनना और फिर वांछित फ़ीचर्स को स्क्वेयर ब्रैकेट्स (बड़े कोष्ठकों) के अंदर शामिल करना भी चाहते हैं।

PYTHON	विवरण
ds["state"]	<ul style="list-style-type: none"> <li>◆ ds: Kickstarter डेटासेट को स्टोर करने वाली चर राशि।</li> <li>◆ []: इस सिम्बल का उपयोग हमारे डेटासेट से सूचनाएं एक्सेस करने के लिए किया जाता है।</li> <li>◆ "state": हम दिखाते हैं कि हम [] के अंदर किस फ़ीचर को चुनना चाहते हैं। फ़ीचर्स को कोटेशन मार्क्स के अंदर लिखना चाहिए क्योंकि वे एक नाम होते हैं।</li> </ul>

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपको आपके डेटासेट के संपूर्ण डेटा का एक सारांश मिल जाना चाहिए, पर उसमें केवल main\_category और state फ़ीचर्स प्रदर्शित होने चाहिए। यदि आपका कोड ठीक से न चले, तो निम्नलिखित को जांचें:

परिणाम	डीबगिंग के सुझाव
<pre>0 असफल 1 असफल 2 असफल 3 असफल 4 निरस्त ... 378656 निरस्त 378657 असफल 378658 असफल 378659 असफल 378660 असफल</pre>	<ul style="list-style-type: none"> <li>◆ Name 'ds' is not defined: आपकी नोटबुक का हर कोड ब्लॉक अलग-अलग चलता है, कभी-कभी हो सकता है कि आपकी नोटबुक प्रोग्राम में अपना स्थान भूल जाए और आपके लिए आपके सभी कोड ब्लॉक्स को चलाना आवश्यक कर दे। यह करने के लिए, अपनी नोटबुक में सबसे ऊपर दिए गए दो तीरों वाले बटन को क्लिक करके सारे कोड ब्लॉक्स चलाएं।</li> <li>◆ क्या आपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा? जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो। कोड की सारी पंक्तियों टाइप करना न भूलें क्योंकि कोडिंग करते समय कोटेशन मार्क्स सही करेक्टर के साथ कॉपी नहीं होते हैं।</li> <li>◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? जांचें कि आपने फ़ीचर्स के नामों की स्पेलिंग सही लिखी हो। याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।</li> <li>◆ क्या आपने अतिरिक्त स्क्वेयर ब्रैकेट्स [] टाइप कर दिए हैं। आप देखेंगे कि जब आप [] सिम्बल टाइप करते हैं तो नोटबुक अपने-आप क्लोज़िंग ब्रैकेट ] जोड़ देती है। इसलिए हो सकता है कि आप गलती से अतिरिक्त ब्रैकेट टाइप कर जाएं। आपके इस कोड में ब्रैकेट्स के कुल दो सेट होने चाहिए: एक सेट के अंदर फ़ीचर्स के नाम हों, और दूसरे सेट के अंदर फ़ीचर्स की लिस्ट हो।</li> </ul>

## चरण 5: plot() मेथड से मिलें (2 मिनट)

pandas लाइब्रेरी, Python की एक शक्तिशाली लाइब्रेरी है जो डेटासेट की जानकारी का विश्लेषण करने, उसमें जोड़-तोड़ यानि मैन्युलेशन करने, और उसे विजुअलाइज़ करने तक के लिए टूल्स प्रदान करती है। तो पेश-ए-खिदमत है pandas में उपलब्ध [plot\(\)](#) मेथड।

PYTHON	विवरण
<code>ds.plot()</code>	<ul style="list-style-type: none"> <li>◆ <code>ds</code>: Kickstarter डेटासेट को स्टोर करने वाली चर राशि।</li> <li>◆ <code>plot()</code>: pandas में उपलब्ध यह मेथड डेटासेट से सूचनाएं प्राप्त करता है और उसे एक ग्राफ के रूप में दिखाता है। हम <code>plot()</code> मेथड का उपयोग हमारे चर राशि <code>ds</code> पर करते हैं, न कि पूरी pandas लाइब्रेरी पर। ऐसा इसलिए है क्योंकि हम हमारे डेटासेट चर राशि में स्टोर की गई जानकारी को प्लॉट करना चाहते हैं।</li> </ul>

`plot()` मेथड एक बेहद शक्तिशाली मेथड है जो हमारे डेटासेट में स्टोर किए गए डेटा को लेकर उसे विभिन्न ग्राफ में बदलने में मदद करता है, जैसे बार ग्राफ, लाइन ग्राफ, बॉक्स प्लॉट, स्कैटर प्लॉट, हिस्टोग्राम एवं कई अन्य रूप। ग्राफ का प्रकार निर्दिष्ट करने और कौनसा डेटा प्रदर्शित होना चाहिए इस बात की पहचान करने के लिए, `plot()` मेथड कई प्रकार के भिन्न-भिन्न पैरामीटर्स, या इनपुट्स स्वीकार करता है। इस गतिविधि में हम इनमें से कुछ पैरामीटर्स के उपयोग पर प्रकाश डालेंगे, विशेष रूप से `kind` एट्रिब्यूट के उपयोग पर। `kind` पैरामीटर एक स्ट्रिंग (अक्षरों का समूह) या शब्द स्वीकारता है, यह उसका इनपुट होता है जो कंप्यूटर को बताता है कि हम किस प्रकार के ग्राफ का प्रदर्शन चाहते हैं। कुछ संभावित इनपुट वैल्यूज में 'pie', 'bar', 'scatter', आदि शामिल हैं। `plot()` मेथड जो विभिन्न प्रकार के इनपुट्स स्वीकारता है उनके बारे में और जानने के लिए आप [pandas के आधिकारिक डॉक्यूमेंटेशन](#) में खोजबीन कर सकते हैं। हम श्रेणीगत और संख्यात्मक डेटा को ग्राफ का रूप देने के लिए आगे के चरणों में इस मेथड का उपयोग करेंगे।

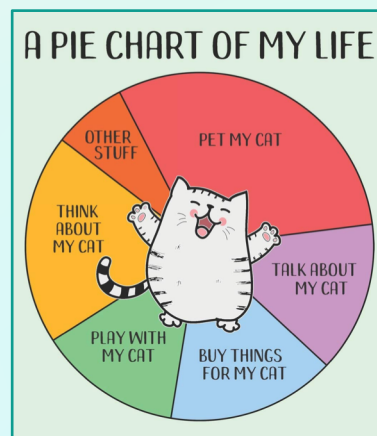
## चरण 6: श्रेणीगत डेटा प्रदर्शित करना (20-30 मिनट)

याद करें कि **श्रेणीगत डेटा** ऐसी सूचना होता है जिसे श्रेणियों में समूहबद्ध किया जा सकता है। इस अनुभाग में हम ऐसे विभिन्न प्रकार के ग्राफ पर गौर करेंगे जो श्रेणीगत डेटा को सर्वोत्तम ढंग से दर्शा सकते हैं। हम प्रत्येक ग्राफ की मुख्य विशेषताओं पर प्रकाश डालेंगे और फिर `plot()` मेथड का उपयोग करके उसे Python में कोड करेंगे।

### पाई चार्ट (2 मिनट)

**पाई चार्ट** गोले की आकृति वाला एक प्रकार का ग्राफ होता है जिसमें डेटा, गोले का एक निश्चित प्रतिशत घेरता है और उसे किसी स्लाइस (जैसे पिज़्ज़ा स्लाइस!) के रूप में दर्शाया जाता है। आइए पाई चार्ट के घटकों को छोटे-छोटे टुकड़ों में समझें।

- ◆ **शीर्षक**: आमतौर पर ग्राफ के ऊपर होता है और हमें बताता है कि किस चीज़ का वर्णन किया जा रहा है।
- ◆ **स्लाइस**: हर स्लाइस संपूर्ण डेटासेट की तुलना में किसी डेटा विशिष्ट के प्रतिशत को निरूपित करती है। कभी-कभी प्रत्येक स्लाइस के बगल में प्रतिशत भी लिखा जाता है।
- ◆ **रिवायत**: यह हमें बताता है कि कौनसा रंग, किस डेटा वैल्यू को निरूपित करता है। कभी-कभी यह ग्राफ की साइड में एक लिस्ट के रूप में मौजूद होता है, या हर स्लाइस के बगल में अलग-अलग लेबल के रूप में दिखता है।



तस्वीर स्रोत: [अमेज़न](#)

### पाई चार्ट चुनते समय विचारणीय प्रश्न:

- ◆ क्या आपके पास श्रेणीगत डेटा है?
- ◆ क्या किसी श्रेणी के डेटा बिंदुओं की संख्या की संपूर्ण डेटासेट से तुलना करना महत्वपूर्ण है?
- ◆ क्या आपके पास बहुत सारी श्रेणियां हैं? हर स्लाइस के आकार पर विचार करें। यदि स्लाइसेज़ की संख्या बहुत अधिक हुई तो पाई चार्ट को पढ़ना मुश्किल हो सकता है।



## Pandas में पाई चार्ट बनाना (5-8 मिनट)

Kaggle में बैंक अप यॉर नोटबुक खोलें। हम आपको बताएंगे कि `kind` एट्रिब्यूट के साथ पाई चार्ट दर्शाने के लिए आप pandas में मौजूद `plot()` मेथड का उपयोग किस प्रकार कर सकते हैं। हमारा Kickstarter डेटासेट एक रो (पंक्ति) में एक प्रोजेक्ट की जानकारी दर्शाता है, पर इसमें ऐसे विशिष्ट रो (पंक्तियाँ) नहीं हैं जो हमें हमारे डेटासेट के सभी प्रोजेक्ट्स के बारे में बताते हों। उदाहरण के लिए, यदि हम फ़िल्म एंड वीडियो के रूप में वर्गीकृत प्रोजेक्ट्स की संख्या जानना चाहते हों तो? या प्रत्येक अवस्था/स्टेट (state) में प्रोजेक्ट्स की संख्या को? यह जानकारी पाने के लिए हम `value_counts()` मेथड का उपयोग करेंगे जो हमें किसी फ़ीचर विशेष में प्रोजेक्ट्स की स्टेट-वार संख्या बताता है। `value_counts()` मेथड से हमें वैल्यूज़ को कुछ फ़ीचर्स के आधार पर समूहबद्ध करने में मदद मिलती है, जैसे, प्रत्येक श्रेणी (कैटेगरी) में प्रोजेक्ट्स की संख्या ज्ञात करना, ताकि हम संपूर्ण डेटासेट का निरूपण करने वाला ग्राफ दर्शाने के लिए `plot()` मेथड का उपयोग कर सकें।

इस उदाहरण में हम आपको बताएंगे कि अलग-अलग `state` में प्रोजेक्ट्स की संख्या दर्शाने के लिए पाई चार्ट कैसे बनाएं। `state` फ़ीचर, Kickstarter प्रोजेक्ट की वर्तमान स्थिति बताता है: `successful` (सफल), `failed` (विफल), `suspended` (स्थगित), `cancelled` (रद्द), `live` (लाइव), या `undefined` (अपरिभाषित)।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें। या तो अपना माउस कर्सर अपने अंतिम कोड ब्लॉक के नीचे ले जाएं और `+Code` बटन दबा दें, या फिर अपनी नोटबुक के ऊपरी बायें भाग में मौजूद `+` बटन दबा दें।
- डेटासेट से `state` फ़ीचर चुनें। याद रखें कि हमने हमारे डेटासेट के रेफ़रेंस को `ds` नामक चर राशि में स्टोर किया है। अब हम `"state"` फ़ीचर को स्क्वेयर ब्रैकेट्स `[]` सिम्बल के अंदर रखकर यह दर्शाते हैं कि हम डेटासेट से केवल इसी कॉलम को चुनना चाहते हैं।
- फ़िल्टर किए हुए डेटासेट पर `value_counts()` मेथड का उपयोग करें।

PYTHON	विवरण
<code>ds["state"].value_counts()</code>	<ul style="list-style-type: none"> <li>◆ <code>ds["state"]</code>: हम <code>state</code> के आधार पर प्रोजेक्ट्स की अलग-अलग संख्या जानना चाहते हैं। हम <code>[]</code> सिम्बल्स का उपयोग हमारे डेटासेट में केवल <code>state</code> फ़ीचर को चुनने के लिए करते हैं और हम फ़ीचर के नाम को डबल कोटेशन मार्क्स में लिखकर यह बताते हैं कि यह एक नाम है।</li> <li>◆ <code>.</code>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।</li> <li>◆ <code>value_counts()</code>: यह मेथड हर विशिष्ट फ़ीचर वैल्यू से जुड़ी एंटिटीज़ (या रो/पंक्तियों) की संख्या रिटर्न करता है।</li> </ul>

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:

परिणाम	डीबगिंग के सुझाव
<div>असफल 197719</div> <div>सफल 133956</div> <div>निरस्त 38779</div> <div>अपरिभाषित 3562</div> <div>लाइव 2799</div> <div>निलंबित 1846</div>	<ul style="list-style-type: none"> <li>◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?</li> <li>◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?</li> <li>◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है</li> <li>◆ क्या आपने अतिरिक्त स्क्वेयर ब्रैकेट्स <code>[]</code> टाइप कर दिए हैं?</li> </ul>

## चरण 6: श्रेणीगत डेटा प्रदर्शित करना (जारी)

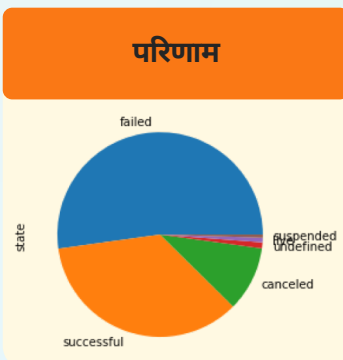
- हमने जो डेटा अभी-अभी आउटपुट किया उसे एक नए चर राशि में स्टोर करें। हमारे state फ्रीचर से अभी-अभी हमें जो डेटा मिला है हम उसे एक पाई चार्ट में बदलने में समर्थ होना चाहते हैं। यदि हम इसे किसी चर राशि में स्टोर कर लें तो यह कहीं आसान हो जाता है। आप अपने चर राशि को कोई भी नाम दे सकते हैं, बस इतना सुनिश्चित करें कि चर राशि का नाम वर्णनकारी हो और camelCase का पालन करता हो। हमारे उदाहरण के लिए, हमने हमारे चर राशि को `projState` नाम दिया है। यदि आपने अपने चर राशि को कोई अन्य नाम दिया है तो इस बात को ध्यान में रखना न भूलें।

```
projState = ds['state'].value_counts()
```

- अपने state डेटासेट पर `plot()` मेथड का उपयोग करें और `kind` पैरामीटर को 'pie' पर सेट कर दें। यह कोड की वह पंक्ति है जो हमारे कंप्यूटर को उस state फ्रीचर डेटा के आधार पर एक पाई चार्ट बनाने को कहती है जिसे हमने अभी-अभी `projState` चर राशि में स्टोर किया था।

PYTHON	विवरण
<pre>projState.plot(kind = 'pie')</pre>	<ul style="list-style-type: none"><li>◆ <code>projState</code>: हम विशिष्ट रूप से प्रोजेक्ट्स की स्टेट-वार संख्याओं (हर स्टेट में प्रोजेक्ट्स की संख्या) को पाई चार्ट के रूप में प्रदर्शित करना चाहते हैं।</li><li>◆ <code>.</code>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।</li><li>◆ <code>plot()</code>: pandas में उपलब्ध यह मेथड डेटासेट से सूचनाएं प्राप्त करता है और उसे एक ग्राफ के रूप में दिखाता है।</li><li>◆ <code>kind = 'pie'</code>: हम <code>kind</code> पैरामीटर (या इनपुट) का उपयोग करते हैं और इसे उस ग्राफ के नाम पर सेट कर देते हैं जो हम दर्शाना चाहते हैं। इस मामले में हम पाई चार्ट दर्शाना चाहते हैं, इसलिए हम वैल्यू को 'pie' पर सेट कर देते हैं। ग्राफ का प्रकार कोटेशन मार्क्स के अंदर होना चाहिए, क्योंकि यह ग्राफ का नाम है।</li></ul>

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



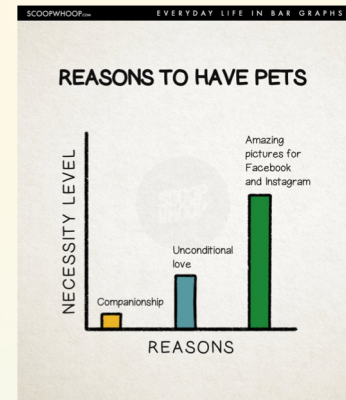
- डीबगिंग के सुझाव**
- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
  - ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
  - ◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
  - ◆ क्या आपने मेथड को कोष्ठकों ( ) के अंदर रखा?

## चरण 6: श्रेणीगत डेटा प्रदर्शित करना (जारी)

### बार ग्राफ (2 मिनट)

**बार ग्राफ** एक ग्राफ होता है जो अलग-अलग ऊंचाइयों वाले बार्स का उपयोग करके डेटा का प्रदर्शन करता है। आइए बार ग्राफ के कुछ घटकों को छोटे-छोटे टुकड़ों में समझें।

- **शीर्षक**
- **बार:** प्रत्येक बार, डेटा की एक श्रेणी को दर्शाता है। बार ग्राफ की ऊंचाई इस बात पर निर्भर करती है कि दूसरे अक्ष पर क्या मापा जा रहा है। बार्स के बीच छोटा सा गैप होना चाहिए।
- **उन्मुखीकरण:** दायीं ओर मौजूद उदाहरणों में हम खड़ी स्थिति में बार्स का उपयोग होता देखते हैं। बार ग्राफ में आड़े पड़े बार्स भी हो सकते हैं।



तस्वीर स्रोत: [ScoopWoop](#)

**पाई चार्ट चुनते समय विचारणीय प्रश्न:**

- ◆ क्या आपके पास श्रेणीगत डेटा है?
- ◆ क्या प्रत्येक श्रेणी (केटेगरी) की सही-सही मात्रा दर्शाना महत्वपूर्ण है? बार ग्राफ प्रत्येक श्रेणी के वास्तविक वैल्यूज़ चित्रित करते हैं, जबकि पाई चार्ट पूरे डेटासेट की तुलना में प्रत्येक श्रेणी का प्रतिशत दर्शाते हैं।

### Pandas में बार ग्राफ बनाना (5-8 मिनट)

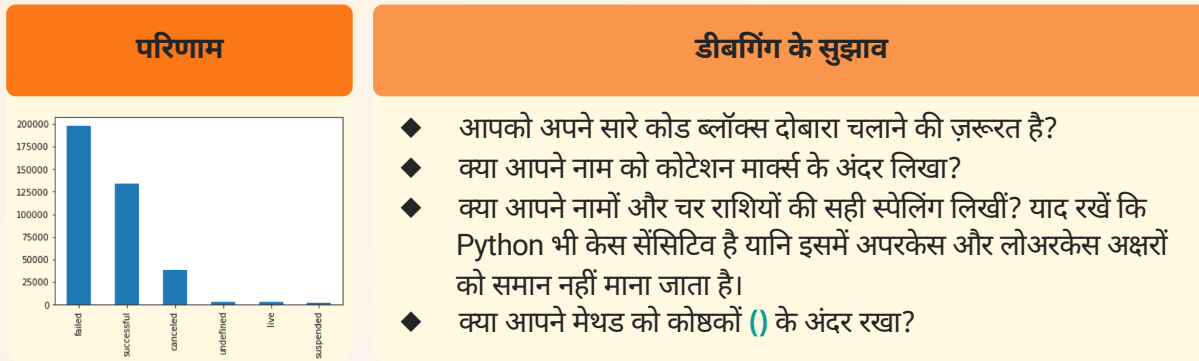
इस उदाहरण में हम Kickstarter प्रोजेक्ट्स के स्टेट (state) दर्शाना जारी रखेंगे, पर इस बार बार ग्राफ के जरिए। इस प्रकार आप पाई चार्ट और बार ग्राफ की आसानी से तुलना कर सकते हैं। इस बार `plot()` मेथड का उपयोग करते समय, हम `kind` पैरामीटर को "bar" पर सेट करेंगे।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- अपने state डेटासेट पर `plot()` मेथड का उपयोग करें और `kind` पैरामीटर को 'bar' पर सेट कर दें। याद रखें कि हमने हर स्टेट (state) में प्रोजेक्ट्स की संख्या से युक्त डेटासेट को `projState` नामक चर राशि में स्टोर किया था। हो सकता है कि आपने इस चर राशि को कोई अलग नाम दिया हो।

PYTHON	विवरण
<pre>projState.plot(kind = 'bar')</pre>	<ul style="list-style-type: none"><li>◆ <code>projState</code>: हम विशिष्ट रूप से प्रोजेक्ट्स की स्टेट-वार संख्याओं (हर स्टेट में प्रोजेक्ट्स की संख्या) को प्रदर्शित करना चाहते हैं।</li><li>◆ <code>.</code>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।</li><li>◆ <code>plot()</code>: pandas में उपलब्ध यह मेथड डेटासेट से सूचनाएं प्राप्त करता है और उसे एक ग्राफ के रूप में दिखाता है।</li><li>◆ <code>kind = 'bar'</code>: हम <code>kind</code> पैरामीटर (या इनपुट) का उपयोग करते हैं और इसे उस ग्राफ के नाम पर सेट कर देते हैं जो हम दर्शाना चाहते हैं। इस मामले में हम बार ग्राफ दर्शाना चाहते हैं, इसलिए हम वैल्यू को 'bar' पर सेट कर देते हैं। ग्राफ का प्रकार कोटेशन मार्क्स के अंदर होना चाहिए, क्योंकि यह ग्राफ का नाम है।</li></ul>

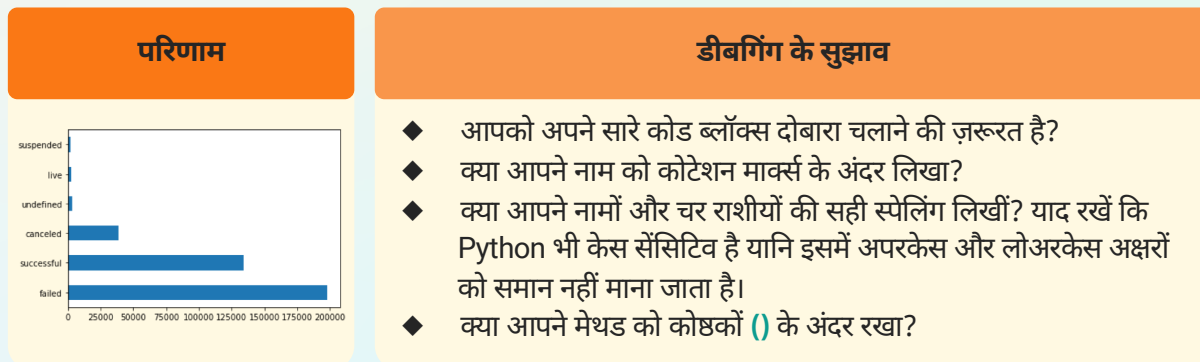
## चरण 6: श्रेणीगत डेटा प्रदर्शित करना (जारी)

- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



इसे आड़े बार ग्राफ में बदलना चाहते हैं? यह आसान है, इस बार `plot()` मेथड का उपयोग करते समय, हम `kind` पैरामीटर को "barh" पर सेट करेंगे।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- अपने state डेटासेट पर `plot()` मेथड का उपयोग करें और `kind` पैरामीटर को 'barh' पर सेट कर दें।
- ```
projState.plot(kind = 'barh')
```
- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



## पाई चार्ट और बार ग्राफ की तुलना (5-8 मिनट)

थोड़ा रुकें और अभी-अभी आपने जो पाई चार्ट और बार ग्राफ बनाए हैं उन पर गौर करें और निम्नलिखित प्रश्न पूछें:

### प्रश्न 1

ग्राफ के आधार पर आप डेटा के बारे में क्या निष्कर्ष निकाल सकते हैं?

### प्रश्न 2

कौनसा ग्राफ आपके निष्कर्ष(ष) पर सर्वाधिक बल देता है?

## चरण 6: श्रेणीगत डेटा प्रदर्शित करना (जारी)

**2 मिनट** लेकर अधिकतम संभव संख्या में वे निष्कर्ष लिखें जो आप अपने बनाए गए ग्राफ के आधार पर निकाल सकते हैं। अपने विचारों को नीचे दी गई टेबल जैसे व्यवस्थित करने से मदद मिल सकती है।

| समापन<br><i>ग्राफ के आधार पर आप डेटा के बारे में क्या निष्कर्ष निकाल सकते हैं?</i> | सर्वोत्तम ग्राफ<br><i>कौन सा ग्राफ आपके निष्कर्ष पर सर्वाधिक बल देता है?</i> |
|------------------------------------------------------------------------------------|------------------------------------------------------------------------------|
|                                                                                    |                                                                              |
|                                                                                    |                                                                              |
|                                                                                    |                                                                              |
|                                                                                    |                                                                              |
|                                                                                    |                                                                              |

नीचे मौजूद हमें मिले कुछ निष्कर्ष देखने से पहले यहां थोड़ा रुकें।

डेटा के बारे में बहुत से निष्कर्ष निकाले जा सकते हैं, आप हमारे कुछ निष्कर्ष पर गौर कर सकते हैं। कौनसा ग्राफ निष्कर्ष को सबसे अच्छे से स्पष्ट करता है इस बारे में आपके उत्तर थोड़े अलग हो सकते हैं और इसमें कुछ भी गलत नहीं है! सबसे महत्वपूर्ण बात यह है कि आप अपने निर्णय को उचित सिद्ध करने में समर्थ हैं।

| समापन<br><i>ग्राफ के आधार पर आप डेटा के बारे में क्या निष्कर्ष निकाल सकते हैं?</i>            | सर्वोत्तम ग्राफ<br><i>कौनसा ग्राफ आपके निष्कर्ष पर सर्वाधिक बल देता है?</i> |
|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------|
| अधिकांश प्रोजेक्ट्स “failed” (विफल) हैं।                                                      | पाई चार्ट                                                                   |
| लगभग 200,000 Kickstarter प्रोजेक्ट्स अंततः “failed” (विफल) state में जा पहुंचे हैं।           | बार ग्राफ                                                                   |
| लगभग 50% Kickstarter प्रोजेक्ट्स, विफल प्रोजेक्ट्स हैं                                        | पाई चार्ट                                                                   |
| undefined (अपरिभाषित), live (लाइव), और suspended (स्थगित) प्रोजेक्ट्स की संख्या लगभग बराबर है | बार ग्राफ                                                                   |

याद रखें कि जब आपको सभी प्रोजेक्ट्स की स्टेट-वार संख्या देखनी हो और यह देखना हो कि प्रतिशत के मामले में वे एक-दूसरे की तुलना में कहां ठहरते हैं, तो ऐसे में पाई चार्ट का उपयोग अच्छा रहता है। प्रोजेक्ट्स की स्टेट-वार संख्या और उनके सही-सही वैल्यूज़ देखने के मामले में बार ग्राफ का उपयोग अच्छा रहता है। आड़े और खड़े बार ग्राफ के बीच का चयन आपकी पसंद-नापसंद पर निर्भर करता है। दोनों ही बिल्कुल समान डेटा दर्शाते हैं, बस विन्यास अलग होता है।



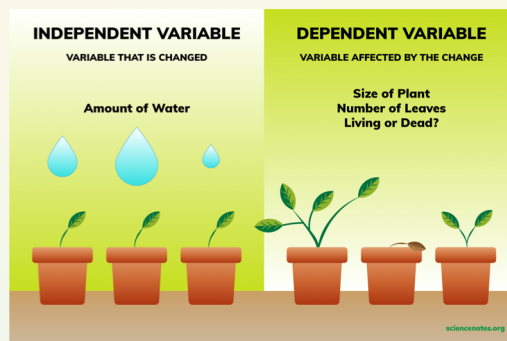
## चरण 7: संख्यात्मक डेटा प्रदर्शित करना (25-35 मिनट)



याद करें कि **संख्यात्मक (न्यूमेरिकल) डेटा** ऐसा डेटा होता है जिसे संख्या रूप में व्यक्त किया जा सकता है। उसकी एक निरंतर रेंज होनी चाहिए (यानि रेंज की सभी संख्याएं निरूपित हुई होनी चाहिए)। चित्रमय (ग्राफिक) संख्यात्मक डेटा पर विचार करते समय, हमें **इंडिपेंडेंट (स्वतंत्र)** और **डिपेंडेंट (आश्रित)** चर राशि की भी पहचान करनी चाहिए।

### इंडिपेंडेंट बनाम डिपेंडेंट चर राशि (2 मिनट)

ग्राफ दो चर राशियों के बीच की तुलना का चित्रण करता है। उदाहरण के लिए, हमने हमारे पाई चार्ट और बार ग्राफ के साथ जो ग्राफ बनाए थे वे Kickstarter प्रोजेक्ट्स की संख्या और उनके स्टेट (अवस्था) की तुलना करते हैं। जिन दो चर राशियों की तुलना हो रही है उनमें से एक को हम **इंडिपेंडेंट चर राशि** और दूसरे को **डिपेंडेंट चर राशि** कहते हैं। **इंडिपेंडेंट चर राशि** वह डेटा होता है जिसमें बदलाव की मात्रा नियंत्रित या पूर्वानुमान-योग्य होती है। हमारे पिछले उदाहरण में विभिन्न प्रकार के स्टेट (जैसे, live (लाइव), failed (विफल), suspended (स्थगित), आदि) वह चर राशि या डेटा हैं जो अपरिवर्तित रहता है। **डिपेंडेंट चर राशि** वह वैल्यू होता है जो इंडिपेंडेंट चर राशि के वैल्यू पर *आश्रित रहते हुए* बदलता है।



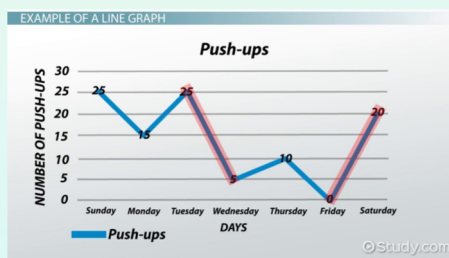
तस्वीर स्रोत: [साइंस नोट्स](#)

हमारे उदाहरण में, प्रोजेक्ट्स की संख्या डिपेंडेंट वेरिएबल होगी क्योंकि वह प्रोजेक्ट के स्टेट (अवस्था) पर आश्रित होती है।

न्यूमेरिकल वैल्यूज़ (संख्यात्मक मानों) की तुलना करते समय, इंडिपेंडेंट चर राशि को **x-अक्ष** या क्षैतिज अक्ष पर दर्शाया जाता है, और डिपेंडेंट चर राशि को **y-अक्ष** या ऊर्ध्व अक्ष पर दर्शाया जाता है। हमें pandas में न्यूमेरिकल वैल्यूज़ को ग्राफ द्वारा दिखाते समय ये फ़ील्ड्स निर्दिष्ट करने होते हैं।

### लाइन ग्राफ (2 मिनट)

**लाइन ग्राफ** ऐसा ग्राफ होता है जो समय के साथ बदलने वाले डेटा को दर्शाता है। लाइन ग्राफ विशिष्ट डेटा पॉइंट्स को दर्शाने के लिए बिंदुओं का उपयोग करता है जिन्हें एक लाइन (रेखा) से जोड़कर समय के साथ ट्रेंड का प्रदर्शन किया जाता है। लाइन ग्राफ का उपयोग श्रेणीगत और संख्यात्मक, **दोनों प्रकार** के डेटा को प्रदर्शित करने के लिए किया जा सकता है। लाइन ग्राफ के कुछ मुख्य घटक होते हैं:



तस्वीर स्रोत: [Study.com](#)

#### ◆ शीर्षक

◆ **X-अक्ष:** इस क्षैतिज अक्ष पर हम डेटा का **इंडिपेंडेंट वेरिएबल (स्वतंत्र चर राशि)** दर्शाते हैं।

◆ **Y-अक्ष:** इस ऊर्ध्व अक्ष पर हम डेटा का **डिपेंडेंट वेरिएबल (आश्रित चर राशि)** दर्शाते हैं।

◆ **ट्रेंड:** लाइन ग्राफ पर, हर डेटा बिंदु एक रेखा (लाइन) से जुड़ा होता है। इससे उतार और चढ़ाव को आसानी से देखने के लिए डेटा के ट्रेंड पर प्रकाश डालने में मदद मिलती है। ऊपर वाले ग्राफ में हम पाते हैं कि जिन दिनों ट्रेंड नीचे की ओर है, उन दिनों लाइन को लाल रंग से हाइलाइट किया गया है। कभी-कभी आप ऐसे लाइन ग्राफ देखेंगे जो रंगों का उपयोग करके ट्रेंड्स के बीच के अंतर दिखाते हैं, या कभी-कभी पूरा ग्राफ एक ही रंग का होता है।

### लाइन ग्राफ चुनते समय विचारणीय प्रश्न:

- ◆ क्या आपके पास एक ऐसा स्वतंत्र चर राशि (इंडिपेंडेंट वेरिएबल) है जो समय मापता है? (जैसे, सप्ताह में दिन, दिन में घंटे, वर्ष में महीने, घंटे में मिनट)
- ◆ क्या प्रत्येक डेटा पॉइंट के बीच के ट्रेंड को देखना महत्वपूर्ण है?

## Pandas में बार ग्राफ बनाना (5-8 मिनट)

थोड़ा रुकें और सोचें कि हमारे Kickstarter डेटासेट में *समय* के निरूपण के लिए किस फ़ीचर का उपयोग किया जा सकता है। `launched` और `deadline`, इन दोनों फ़ीचर्स को समय में मापा जाता है! इस उदाहरण के लिए हम Kickstarter प्रोजेक्ट्स की `goal` राशि की तुलना, `launched` के दिनांक से करेंगे। याद रखें कि हमें इस बात पर विचार करना है कि कौनसा फ़ीचर हमारा इंडिपेंडेंट चर राशि है और कौनसा फ़ीचर हमारा डिपेंडेंट चर राशि है। नीचे हमारा उत्तर देखने से पहले स्वयं सोचें।

- ◆ **ग्राफ:** Kickstarter प्रोजेक्ट्स के लॉन्च दिनांक और गोल (लक्ष्य) राशि की तुलना
- ◆ **इंडिपेंडेंट चर राशि:** `deadline`. यहां समय अपरिवर्तित रहता है!
- ◆ **डिपेंडेंट चर राशि:** `goal` (लक्ष्य)। कोई Kickstarter प्रोजेक्ट अपने गोल (लक्ष्य) के रूप में जो राशि तय करता है वह समय पर निर्भर करते हुए अलग-अलग हो सकती है, इसलिए यही हमारा डिपेंडेंट चर राशि है।

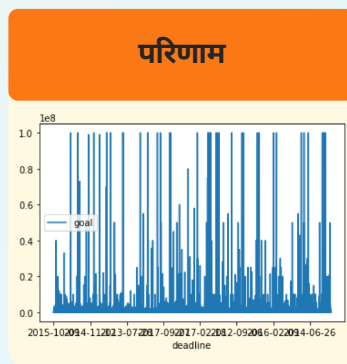
इस बार `plot()` मेथड का उपयोग करते समय, हमें `x` और `y` पैरामीटर्स को उन फ़ीचर्स पर सेट करना होगा जिन्हें हम दिखाना चाहते हैं, और हमें `kind` पैरामीटर को `"line"` पर सेट करना होगा।

`plot()` मेथड डिफ़ॉल्ट रूप से लाइन ग्राफ को चुनता है, इसलिए `kind` पैरामीटर को `"line"` पर सेट करना वैकल्पिक है।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- अपने डेटासेट पर `plot()` मेथड का उपयोग करके निम्नलिखित पैरामीटर्स सेट करें:
  - ◆ `x` पैरामीटर अपने इंडिपेंडेंट चर राशि के लिए।
  - ◆ `y` पैरामीटर अपने डिपेंडेंट चर राशि के लिए।
  - ◆ `kind` पैरामीटर, `'line'` के लिए।

```
ds.plot(x='deadline', y='goal', kind = 'line')
```

- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



**डीबगिंग के सुझाव**

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
- ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
- ◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखीं? याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ क्या आपने मेथड को कोष्ठकों `()` के अंदर रखा?

आप यह देखेंगे कि इस लाइन ग्राफ से हमारे डेटा के बारे में हमें ज़्यादा कुछ पता नहीं चलता है क्योंकि हमारे डेटासेट में इतनी ही जानकारी दर्ज हुई है। इस लाइन ग्राफ ने सभी 378,661 प्रोजेक्ट्स को इस ग्राफ में एक अकेले बिंदु के रूप में प्लॉट कर दिया। यह बहुत ज़्यादा है! हम बस हर ग्राफ के बारे में यह जानने की कोशिश कर रहे हैं कि वह क्या है और उसे pandas में कैसे कोड किया जाए - हमें सारे प्रोजेक्ट्स प्लॉट करने की ज़रूरत नहीं है।

## चरण 7: संख्यात्मक डेटा प्रदर्शित करना (जारी)

चलिए `head()` मेथड का उपयोग सीखते हैं जो हमारे डेटासेट की बस कुछ शुरुआती रो (पंक्तियों) (या एंटिटीज़) का चयन करता है। हम इसका उपयोग करके हमारे डेटासेट के शुरुआती 100 प्रोजेक्ट्स दर्शाएंगे। यहां हम `head()` और `plot()` मेथड को एक साथ समूहबद्ध करने के लिए मेथड चेनिंग के उपयोग का अभ्यास भी करेंगे।

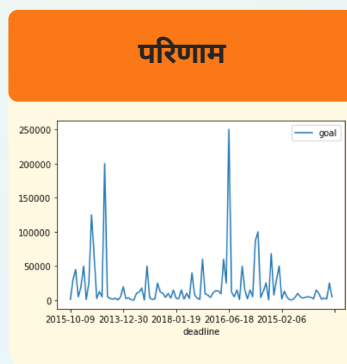
- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- पैरामीटर 100 के साथ `head()` मेथड का उपयोग अपने डेटासेट पर करके केवल शुरुआती 100 प्रोजेक्ट्स प्राप्त करें

| PYTHON                    | विवरण                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ds.head(100)</code> | <ul style="list-style-type: none"><li>◆ <code>ds</code>: यह वह चर राशि है जिसमें हमने हमारा डेटासेट स्टोर किया था।</li><li>◆ <code>.</code>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।</li><li>◆ <code>head()</code>: pandas में मौजूद यह मेथड हमारे डेटासेट की शुरुआती कुछ एंटिटीज़ (या रो/पंक्तियां) प्राप्त करता है।</li><li>◆ <code>100</code>: <code>head()</code> मेथड एक पूर्णांक (या पूर्ण संख्या) को अपने इनपुट के रूप में स्वीकारता है जिससे यह निर्दिष्ट होता है कि कितनी रो (पंक्तियां) चुननी हैं। हमारे उदाहरण में हम केवल शुरुआती 100 रो (पंक्तियां) चाहते हैं।</li></ul> |

- `plot()` मेथड को अपने डेटासेट पर चेन (`chain`) करें और `x`, `y`, तथा `kind` पैरामीटर्स सेट करें:

```
ds.head(100).plot(x='deadline', y='goal', kind = 'line')
```

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



**डीबर्गिंग के सुझाव**

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
- ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
- ◆ क्या आपने नामों और चर राशीयों की सही स्पेलिंग लिखीं? याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ क्या आपने मेथड को कोष्ठकों `()` के अंदर रखा?

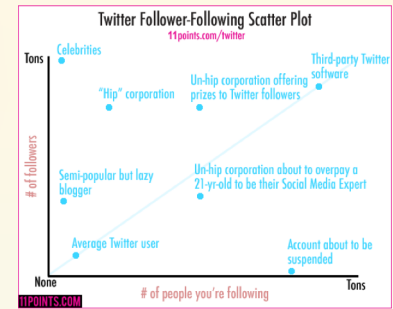
## चरण 7: संख्यात्मक डेटा प्रदर्शित करना (जारी)

### स्कैटर प्लॉट (2 मिनट)

**स्कैटर प्लॉट** एक ग्राफ है जिसमें दो चर राशियों की तुलना करते समय बिंदुओं का उपयोग करके प्रत्येक डेटा पॉइंट दर्शाया जाता है। यह लाइन ग्राफ जैसा होता है जिसमें प्रत्येक डेटा को एक बिंदु से दर्शाया जाता है, पर इसमें बिंदुओं को लाइन से जोड़ा नहीं जाता है। स्कैटर प्लॉट का उपयोग प्रायः ढेर सारे डेटा वाले ग्राफ दर्शाने के लिए किया जाता है, ताकि पता चल सके कि डेटा में कोई ट्रेंड या सहसंबंध है या नहीं। ट्रेंड्स किसी लाइन के रूप में, विभिन्न प्रकार के वक्रों (कर्व) के रूप में या फिर पूरी तरह नामौजूद हो सकते हैं।

आइए स्कैटर प्लॉट के घटकों को छोटे-छोटे टुकड़ों में समझें:

- **शीर्षक**
- **X-अक्ष:** इस क्षैतिज अक्ष पर हम डेटा का **इंडिपेंडेंट वेरिएबल (स्वतंत्र चर राशि)** दर्शाते हैं।
- **Y-अक्ष:** इस ऊर्ध्व अक्ष पर हम डेटा का **डिपेंडेंट वेरिएबल (आश्रित चर राशि)** दर्शाते हैं।
- **ट्रेंड:** अलग-अलग बिंदुओं, या डेटा, को जोड़ा नहीं जाता है। बल्कि प्रायः ट्रेंड ही डेटा को सारांशित या समूहबद्ध करके किसी संबंध विशेष पर प्रकाश डालता है। यदि ट्रेंड का सबसे अच्छा वर्णन एक लाइन के रूप में किया जा सकता हो, तो आपने इसे “द लाइन ऑफ बेस्ट फिट” के नाम से सुना होगा।



तस्वीर स्रोत: [11.](#)

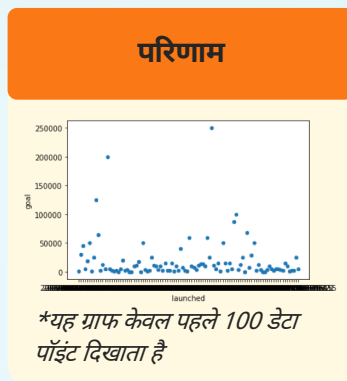
### Pandas में स्कैटर प्लॉट बनाना (3-5 मिनट)

लाइन ग्राफ में हमें जो सबसे बड़ी चुनौतियां मालूम पड़ीं उनमें से एक यह थी कि हर बिंदु को जोड़ने वाली लाइन से डेटा में ट्रेंड देख पाना कठिन हो जाता है। आइए एक बार फिर Kickstarter प्रोजेक्ट के **goal** की तुलना **launched** के दिनांक से करने की कोशिश करते हैं। इस बार **plot()** मेथड का उपयोग करते समय, हमें **x** और **y** पैरामीटर्स को उन फीचर्स पर सेट करना होगा जिन्हें हम दिखाना चाहते हैं, और हमें **kind** पैरामीटर को “**scatter**” पर सेट करना होगा।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- अपने डेटासेट पर **plot()** मेथड का उपयोग करके निम्नलिखित पैरामीटर्स सेट करें:
  - ◆ **x** पैरामीटर अपने इंडिपेंडेंट चर राशि के लिए।
  - ◆ **y** पैरामीटर अपने डिपेंडेंट चर राशि के लिए।
  - ◆ **kind** parameter to ‘**scatter**’.

```
ds.plot(x='deadline', y='goal', kind = 'scatter')
```

- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपके पास नीचे दी गई तालिका जैसा परिणाम होना चाहिए:



### डीबगिंग के सुझाव

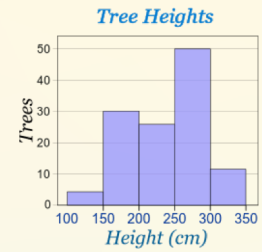
- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
- ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
- ◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ क्या आपने मेथड को कोष्ठकों (**()**) के अंदर रखा?

इस ग्राफ से हम एक निष्कर्ष संभवतः यह निकाल सकते हैं कि अधिकांश Kickstarter प्रोजेक्ट्स के **goal** की वैल्यू 50,000 से कम है।

## चरण 7: संख्यात्मक डेटा प्रदर्शित करना (जारी)

### हिस्टोग्राम (2 मिनट)

**हिस्टोग्राम** एक ग्राफ होता है जो अलग-अलग ऊँचाइयों वाले बार्स का उपयोग करके डेटा दर्शाता है। बार ग्राफ के विपरीत, हिस्टोग्राम डेटा को रेंज (परास) में समूहबद्ध कर देते हैं। यही कारण है कि हिस्टोग्राम का उपयोग केवल **संख्यात्मक डेटा** के साथ किया जा सकता है। आइए हिस्टोग्राम के कुछ घटकों को छोटे-छोटे टुकड़ों में समझें:



तस्वीर स्रोत: मैथ इज़ फ़न

- ◆ **शीर्षक**
- ◆ **बार:** प्रत्येक बार, डेटा में वैल्यूज की एक रेंज दर्शाती है। बार ग्राफ की ऊँचाई उस रेंज के अंदर आने वाले डेटा पॉइंट्स की संख्या को मापती है। बार ग्राफ के विपरीत, बार्स के बीच कोई **खाली स्थान नहीं** होता है।
- ◆ **बिन्स (Bins):** इससे यह तय होता है कि ग्राफ पर कितने बार्स दर्शाए जाने चाहिए। pandas में यह वैल्यू डिफ़ॉल्ट रूप से 10 बिन्स (या बार्स) होती है।

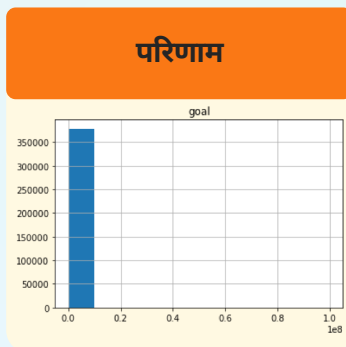
### Pandas में हिस्टोग्राम बनाना (15-20 मिनट)

इस उदाहरण में हम Kickstarter प्रोजेक्ट्स की गोल (लक्ष्य) राशियों की रेंज को विभाजित करेंगे। pandas में हिस्टोग्राम बनाने के लिए, हम या तो `plot()` मेथड का उपयोग कर सकते हैं और `kind` पैरामीटर को 'hist' पर सेट कर सकते हैं, या फिर हम `hist()` मेथड का उपयोग कर सकते हैं। दोनों मेथड्स बिल्कुल एक जैसा कार्य करते हैं, पर `hist()` मेथड में ऐसे विशिष्ट पैरामीटर्स शामिल हैं जिनसे हमें ऐसे फ़ीचर्स को कस्टमाइज़ करने की सुविधा मिलती है जो केवल हिस्टोग्राम में ही मिलते हैं, जैसे कि `bins` की संख्या।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- अपने डेटासेट पर `hist()` मेथड का उपयोग करें और `column` पैरामीटर को 'goal' फ़ीचर पर सेट कर दें।

| PYTHON                              | विवरण                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>ds.hist(column = 'goal')</pre> | <ul style="list-style-type: none"> <li>◆ <code>ds</code>: यह वह चर राशि है जो हमारा डेटासेट स्टोर करता है।</li> <li>◆ <code>.</code>: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।</li> <li>◆ <code>hist()</code>: pandas में यह मेथड, हिस्टोग्राम बनाता है</li> <li>◆ <code>column = 'goal'</code>: हम <code>column</code> पैरामीटर का उपयोग यह निर्दिष्ट करने के लिए करते हैं कि हम किस संख्यात्मक फ़ीचर में वैल्यूज को छोटी-छोटी रेंज में विभाजित देखना चाहते हैं।</li> </ul> |

- अपना कोड ब्लॉक चलाएं।



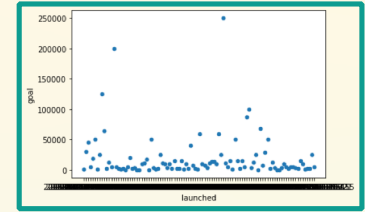
### डीबगिंग के सुझाव

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
- ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
- ◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ क्या आपने मेथड को कोष्ठकों `()` के अंदर रखा?



## चरण 7: संख्यात्मक डेटा प्रदर्शित करना (जारी)

आप सोच रहे होंगे कि आपको इस हिस्टोग्राम में केवल एक बार क्यों दिख रहा है। `hist()` का उपयोग करते समय `bins` की डिफ़ॉल्ट वैल्यू 10 होती है। इसका अर्थ है कि हमें ठीक-ठीक 10 बार दिखने चाहिए, पर हमें केवल एक ही क्यों दिख रहा है? याद करें हमारे स्कैटर प्लॉट को जिस में हमने समय के साथ प्रोजेक्ट्स के गोल (लक्ष्य) की तुलना की थी। अधिकांश प्रोजेक्ट्स की गोल (लक्ष्य) वैल्यू 50,000 या इससे कम थी, पर कुछ ऐसे प्रोजेक्ट्स भी थे जिनका प्रोजेक्ट गोल (लक्ष्य)



100,000,000 जितने उच्च और 0.01 जितने निम्न थे। जब pandas इस रेंज को 10 बिन में बांटने की कोशिश करती है तो हर बिन की रेंज बहुत विशाल होती है। अधिकांश प्रोजेक्ट्स पहले बिन में आते हैं जिसे हिस्टोग्राम में चित्रित किया गया है। अन्य रेंज में भी प्रोजेक्ट्स मौजूद हैं, पर चूंकि वे बहुत छोटे हैं इसलिए वे इस हिस्टोग्राम में नामौजूद मालूम पड़ते हैं। चलिए थोड़ी सी **डेटा की जोड़-तोड़** करके 5000 और इससे कम गोल (लक्ष्य) वाले प्रोजेक्ट्स देखते हैं। याद रखें कि pandas में हम `[]` सिम्बल्स का उपयोग कर सकते हैं और अंदर फ़िल्टर जोड़ सकते हैं, जैसे कि तब जब हम केवल कुछ फ़ीचर्स पर प्रकाश डालना चाहते थे। इस मामले में हम एक **कंडीशनल** जोड़ कर 5000 से कम के `goal` वैल्यूज़ प्राप्त करते हैं।

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- डेटासेट के `goal` फ़ीचर पर एक ऐसी कंडीशन (शर्त) लिखें जो केवल 5000 से कम वाले वैल्यूज़ प्राप्त करती हो।

| PYTHON                              | विवरण                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>(ds["goal"] &lt; 5000)</code> | <ul style="list-style-type: none"><li>◆ <code>ds["goal"]</code>: हम हमारे डेटासेट को फ़िल्टर करना चाहते हैं, हम हमारे डेटासेट चर राशि <code>ds</code> का उपयोग करते हैं, और फिर स्क्वेयर ब्रैकेट्स <code>[]</code> का उपयोग करके <code>"goal"</code> (लक्ष्य) फ़ीचर पर फ़िल्टरिंग निर्दिष्ट करते हैं।</li><li>◆ <code>&lt; 5000</code>: चूंकि हम केवल 5000 से कम गोल (लक्ष्य) वैल्यू वाले प्रोजेक्ट्स चाहते हैं।</li><li>◆ <code>()</code>: हम कंडीशनल को कोष्ठकों के अंदर रखते हैं ताकि कंप्यूटर कंडीशनल को अलग पहचान सके।</li></ul> |

- आपने जो कंडीशनल लिखा था उसे `[]` के अंदर डेटासेट पर लागू करें। यहां हम पिछले चरण वाले कंडीशनल स्टेटमेंट्स लागू करने के लिए `ds` चर राशि का और `[]` सिम्बल्स का उपयोग करेंगे। स्क्वेयर ब्रैकेट्स `[]` के अंदर कंडीशनल स्टेटमेंट (सशर्त कथन) जोड़ कर हम कंप्यूटर को बताते हैं कि वह `ds` डेटासेट में ऐसी सभी एंट्रीज़ (या रो/पंक्तियां) खोजे जहां `'goal'` फ़ीचर का वैल्यू 5000 से कम हो।

```
ds[(ds['goal'] < 5000)]
```

- इस नए-नए फ़िल्टर हुए डेटासेट को एक नए चर राशि में स्टोर करें। हम इस फ़िल्टर किए हुए डेटासेट को एक नए चर राशि में स्टोर करना चाहते हैं ताकि हम मूल डेटासेट को संशोधित न कर दें। हमारे उदाहरण में हमने इस चर राशि को `under5000` नाम दिया है।

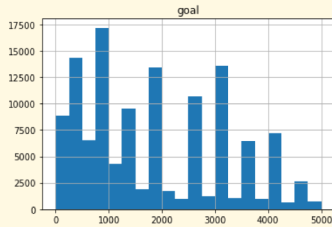
```
under5000 = ds[(ds['goal'] < 5000)]
```

- अपने फ़िल्टर किए हुए डेटासेट पर `hist()` मेथड का उपयोग करें और `column` पैरामीटर को `'goal'` फ़ीचर पर सेट कर दें।

```
under5000.hist(column='goal')
```

→ अपना कोड ब्लॉक चलाएं।

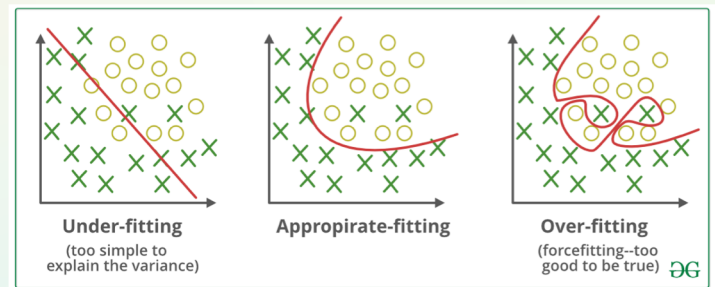
### परिणाम



### डीबगिंग के सुझाव

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत है?
- ◆ क्या आपने नाम को कोटेशन मार्क्स के अंदर लिखा?
- ◆ क्या आपने नामों और चर राशियों की सही स्पेलिंग लिखी? याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ क्या आपने मेथड को कोष्ठकों ( ) के अंदर रखा?
- ◆ क्या आपने डेटासेट को स्क्वेयर ब्रैकेट्स [ ] के अंदर रखा?
- ◆ क्या आपने कॉमा का उपयोग करके पैरामीटर्स को अलग किया?

हमारे ग्राफ के बिन्स की संख्या बढ़ाकर हम देख सकते हैं कि विभिन्न गोल (लक्ष्य) राशियों पर प्रोजेक्ट्स के शिखर हैं। जब हमने 10 बिन्स का उपयोग किया था तब यह बात इतनी स्पष्ट नहीं थी। हम bin जैसे पैरामीटर्स के साथ प्रयोग करके पता कर सकते हैं कि कौनसे पैरामीटर्स डेटा के साथ सबसे अच्छी तरह फिट होते हैं। इस प्रक्रिया को **फ़िटिंग** कहते हैं। **ओवरफ़िटिंग**, यानी जहां उल्लेखनीय रूप से भिन्न प्रेक्षणों को शामिल किया गया हो वहां बहुत सारे डेटा से सहसंबंध स्थापित करने की कोशिश करना, और **अंडरफ़िटिंग**, यानी जहां ट्रेंड को शुद्धतापूर्वक नहीं मापा गया है वहां पर्याप्त डेटा से सहसंबंध स्थापित नहीं करना, के बीच एक नाज़ुक संतुलन होता है।



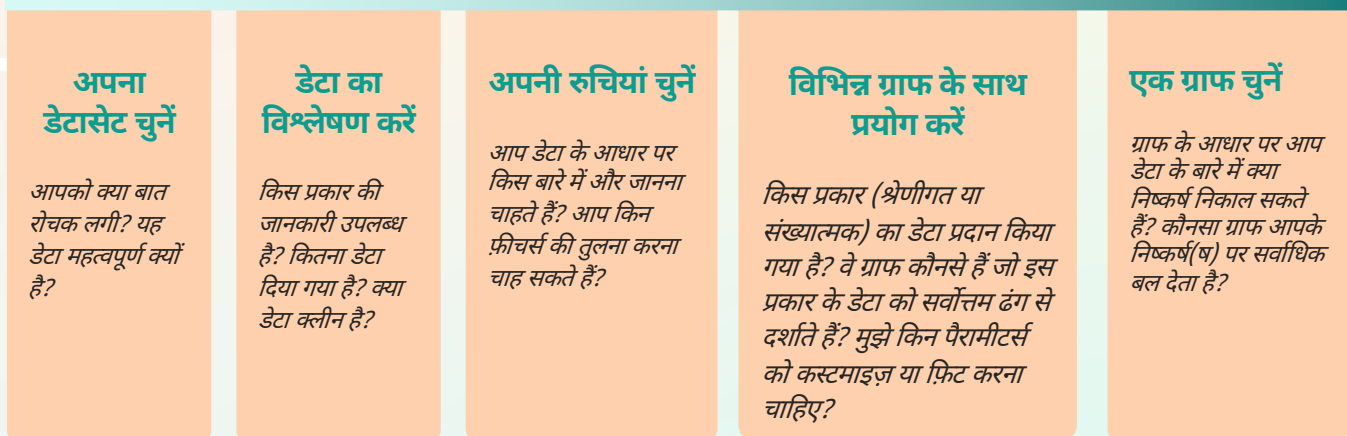
तस्वीर स्रोत: [GeeksforGeeks](https://www.geeksforgeeks.org/)

## चरण 8: विस्तार (5-40 मिनट)

### एक्सटेंशन 1: अन्य Kaggle डेटासेट्स में खोजबीन (30-40 मिनट)

Kaggle पर बहुत से डेटासेट उपलब्ध हैं। थोड़ा समय निकाल कर [यहां](#) कुछ डेटासेट्स में खोजबीन करें। आप **यूज़ेबिलिटी (उपयोज्यता)** के अनुसार परिणामों को क्रमबद्ध करना चाह सकते हैं। इस स्कोर से आपको यह पता करने में मदद मिलती है कि कोई डेटासेट कितना “क्लीन” है, या उपयोग एवं व्याख्या के लिए कितना तैयार है।

किसी नए डेटासेट का विश्लेषण करते समय, आप शुरुआत करने से पहले डेटा को गहराई से समझना सुनिश्चित करना चाह सकते हैं। यहां कुछ चरण दिए जा रहे हैं जिनका आप पालन कर सकते हैं।



जब आप अपना डेटासेट चुन चुके हों और डेटा को विज़ुअलाइज़ (दृष्टिगोचर) कर सकने के तरीकों पर विचार-मंथन कर चुके हों, तो यह अपना विश्लेषण शुरू करने का समय होता है! इस गतिविधि में हमने ऐसी कुछ चीज़ों को मात्र सतही तौर पर जाना है जिनसे [pandas](#) डेटा विज़ुअलाइज़ेशन में मदद कर सकती है, पर [pandas](#) और भी बहुत कुछ कर सकती है! इस शक्तिशाली लाइब्रेरी के बारे में और जानने के लिए इन संसाधनों पर नज़र डालें!

- ◆ [Pandas का दस्तावेज़ीकरण: मानसिक चित्रण](#)
- ◆ [Real Python: प्लॉट विद Pandas](#)

हो सकता है कि आप यह पाएं कि अपने विश्लेषण को और आगे ले जाने के लिए आपके डेटा को थोड़ा अपडेट या मेनिपुलेट करने की ज़रूरत है। हम Python में डेटा की जोड़-तोड़ (मेनिपुलेशन) से संबंधित हमारी [डेटा प्लेग्राउंड](#) गतिविधि को देखने की पुरज़ोर सिफ़ारिश करते हैं।

## एक्सटेंशन 2: अपने ग्राफ को कस्टमाइज़ करना (5-15 मिनट)

**plot()** मेथड में कई अलग-अलग पैरामीटर्स होते हैं। आइए थोड़ा रुकें और उन कुछ पैरामीटर्स पर प्रकाश डालें जो शामिल करने लायक रोचक हो सकते हैं।

- ◆ **title (शीर्षक):** यह पैरामीटर एक String (अक्षर-समूह) या शब्द स्वीकारता है, और उसे ग्राफ के ऊपर शीर्षक रूप में दिखाता है। यदि आपके पास एक से अधिक प्लॉट (ग्राफ) हों, तो आप शीर्षकों की पूरी सूची दे सकते हैं और pandas शीर्षकों को तदनुसार अलग कर लेगी।
- ◆ **grid (ग्रिड):** यह पैरामीटर आपके प्लॉट में ग्रिड लाइन्स जोड़ देता है। ग्रिड लाइन्स जोड़ने के लिए, आपको इस पैरामीटर को True पर सेट करना होगा।
- ◆ **legend (रिवायत):** इस पैरामीटर को True पर सेट करके अपने ग्राफ के साइड में लेजेंड (कुंजी) जोड़ सकते हैं
- ◆ **xlim:** pandas आपके ग्राफ को वैल्यूज़ की सर्वोत्तम रेंज के अनुसार अपने-आप कॉन्फ़िगर कर देती है। आप इस पैरामीटर का उपयोग अपने ग्राफ के क्षैतिज अक्ष पर वैल्यूज़ की रेंज को सीमित करने के लिए कर सकते हैं। आपको कोष्ठकों का उपयोग करते हुए क्रमबद्ध जोड़े के रूप में रेंज प्रविष्ट करनी होगी।
- ◆ **ylim:** xlim की तरह ylim का उपयोग आपके ग्राफ के ऊर्ध्व अक्ष पर वैल्यूज़ की रेंज को सीमित करने के लिए किया जा सकता है।
- ◆ **xlabel:** यह पैरामीटर एक String (अक्षरसमूह) स्वीकारता है और उसे क्षैतिज अक्ष पर लेबल रूप में जोड़ देता है।
- ◆ **ylabel:** यह पैरामीटर एक String (अक्षरसमूह) स्वीकारता है और उसे ऊर्ध्व अक्ष पर लेबल रूप में जोड़ देता है।
- ◆ **color:** यह पैरामीटर एक String (अक्षरसमूह) स्वीकारता है और आपके ग्राफ में डेटा का रंग बदल देता है। आप नामित रंगों की सूची [यहां](#) देख सकते हैं।

अपने ग्राफ के लुक को कस्टमाइज़ करने के लिए इनमें से कुछ पैरामीटर्स के साथ प्रयोग करके देखें।

## एक्सटेंशन 3: सीबॉर्न (Seaborn) की खोजबीन (20-30 मिनट)

Pandas एक शक्तिशाली टूल है जिसका उपयोग प्रोग्रामर्स द्वारा डेटा खोजने, फ़िल्टर करने, तुलना करने, संशोधित करने और विज़ुअलाइज़ करने के लिए किया जाता है। हालांकि, आपने देखा होगा कि कुछ ग्राफ ज़्यादा सुंदर नहीं दिखते हैं। पेश है एक और Python लाइब्रेरी, **Seaborn**। Seaborn एक ऐसी लाइब्रेरी है जो विशेष रूप से डेटा विज़ुअलाइज़ेशन के लिए है, यानी pandas के विपरीत, इसके अधिकांश फ़ंक्शन और एट्रिब्यूट ग्राफ को अधिक व्यवस्थित दिखाने में और प्रोग्रामर्स को अतिरिक्त कस्टमाइज़ेशन की योग्यता प्रदान करने में विशेषज्ञता रखते हैं।

आपकी Kaggle नोटबुक में seaborn का उपयोग करने के लिए हमें पहले लाइब्रेरी को इम्पोर्ट करना होगा।

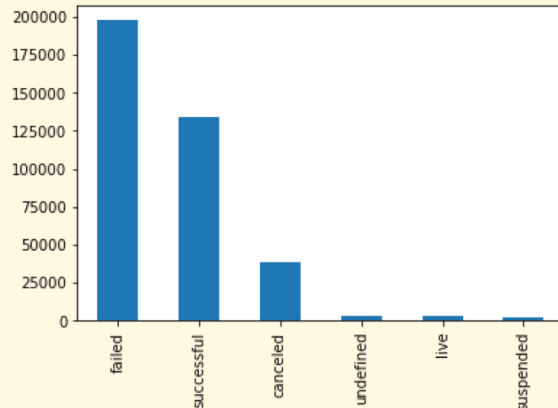
| PYTHON                             | विवरण                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>import seaborn as sns</code> | <ul style="list-style-type: none"> <li>◆ <b>import:</b> यह कीवर्ड कंप्यूटर को बताता है कि हम एक Python लाइब्रेरी का उपयोग कर रहे हैं।</li> <li>◆ <b>as:</b> यह कीवर्ड, पैकेज को एक उपनाम देता है। यह एक वैकल्पिक चरण है पर इससे प्रोग्रामिंग आसान हो सकती है।</li> <li>◆ <b>seaborn/sns:</b> इस Python लाइब्रेरी का उपयोग डेटा को विज़ुअलाइज़ करने में किया जाता है। हमने इस पैकेज को <b>sns</b> उपनाम दिया है।</li> </ul> |

## चरण 8: विस्तार (जारी)

आइए प्रत्येक स्टेट (state) में प्रोजेक्ट्स की संख्या को चित्रित करने के लिए हमारे बनाए कुछ बार ग्राफ्स की तुलना करते हैं:

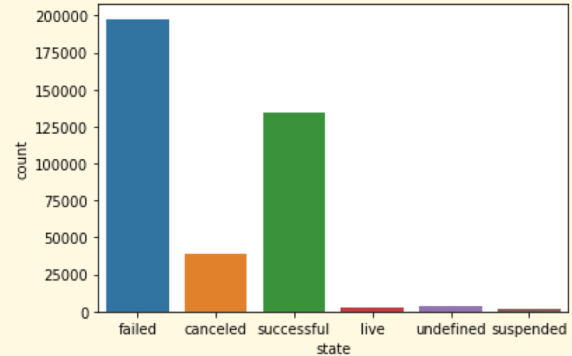
### PANDAS

```
projState = ds['state'].value_counts()  
projState.plot(kind='bar')
```



### SEABORN

```
sns.countplot(x='state', data=ds)
```



seaborn लाइब्रेरी का उपयोग करने पर हम देखते हैं कि हर बार का रंग अलग है, x और y अक्ष पर लेबल पहले से हैं, और कम पंक्तियों वाले कोड की ज़रूरत पड़ती है। `countplot()` मेथड एक डेटासेट स्वीकारता है और फ़ीचर में हर अद्वितीय वैल्यू की बारंबारता/आवृत्ति की गणना करता है। इसे करने के लिए pandas का उपयोग करने पर हमें `value_counts()` फ़ंक्शन का उपयोग करना पड़ता था!

थोड़ा समय निकाल कर [seaborn](#) लाइब्रेरी को और इससे मिलने वाले अन्य प्रकार के ग्राफ्स को जानें। यदि आप और जानना चाहते हों तो यहां हम कुछ अतिरिक्त संसाधन भी दे रहे हैं:

- [The Ultimate Python Seaborn ट्यूटोरियल: Gotta Catch 'Em All](#)
- [Data Camp का Python Seaborn Tutorial For Beginner](#)



## चरण 9: अपने Girls Who Code at Home परियोजना को साझा करें! (5-10 मिनट)

हम आपके काम को देखना पसंद करेंगे और हम जानते हैं कि दूसरे भी ऐसा करेंगे। अपनी अंतिम परियोजना को हमारे साथ साझा करें। @girlswhocode #codefromhome को टैग करना मत भूलें, और हो सकता है कि हम आपको हमारे खाते में प्रदर्शित कर देंगे!

### अपना काम सेव करना (2-5 मिनट)

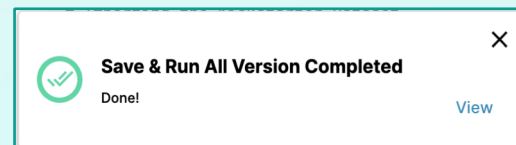
- **Save Version बटन पर क्लिक करें।** आपने अपनी नोटबुक के ऊपरी दायें कोने में **Save Version** बटन देखा होगा। इससे एक नई विंडो खुलनी चाहिए।
- **एक Version Name जोड़ें।** यह वैकल्पिक फ़ील्ड आपके लिए यह दर्ज करने का एक अच्छा तरीका है कि आपने इस नए वर्ज़न (संस्करण) में ऐसा क्या किया है जो पिछले संस्करणों से अलग है। Kaggle अपने-आप आपके संस्करणों को क्रमांक देता जाएगा, इससे पुराने संस्करणों को देखना आसान हो जाता है।
- **Version Type चुनें।** हमारी सलाह है कि आप **Save & Run All** विकल्प चुनें।

◆ **Quick Save:** यह अपने काम को बेहद तेज़ी से सेव करने का एक अच्छा तरीका है। यह संस्करण हर चीज़ को ठीक-ठीक वैसे सेव कर देगा जैसी वह आपकी नोटबुक में प्रदर्शित हो रही है। यदि आपने अपनी नोटबुक में संपादन किए थे पर सभी कोड ब्लॉक्स को दोबारा नहीं चलाया था तो इस संस्करण से समस्याएं पैदा हो सकती हैं।

◆ **Save & Run All:** यह सारे कोड ब्लॉक्स को चलाकर और फिर इस संस्करण को सेव करके आपकी नोटबुक की एक ताज़ा कॉपी सेव कर देता है। यदि आपके पास समय हो तो अपनी नोटबुक्स सेव करने का हमेशा यही सबसे अच्छा तरीका है।



- **Save बटन पर क्लिक करके इंतज़ार करें।** जब आप अपने सेव संस्करण विकल्पों की पुष्टि कर चुके हों, तो **Save** बटन पर क्लिक कर दें। आपको एक पॉप-अप विंडो दिखेगी जो आपके सेव की स्थिति बताएगी। आपके संस्करण को पूरी तरह सेव होने में एक मिनट का समय लग सकता है।



## चरण 9: अपने Girls Who Code at Home परियोजना को साझा करें! (जारी)

### अपना काम शेयर करना (3-5 मिनट)

- **Share आइकन पर क्लिक करें।** आपकी नोटबुक के ऊपरी दाहिने कोने में, Save Version बटन के पास, Share बटन होता है।



- **Privacy को बदलकर Public कर दें।** Privacy फ़ील्ड के दाहिनी ओर वाला ड्रॉप डाउन कॉलम चुनें और उसमें से **Public** चुनें। इससे एक चेतावनी संदेश दिखेगा कि सुनिश्चित करें कि आप इस बात से अवगत हैं कि अन्य उपयोक्ता आपका प्रोजेक्ट देख सकेंगे। **Ok, make public** चुनें।



- **Turn off comments बॉक्स पर निशान लगा दें।** Turn off comments विकल्प के बायीं ओर मौजूद चेकबॉक्स पर क्लिक करें।

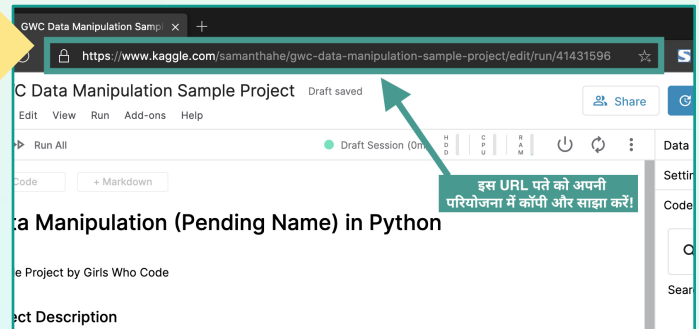
- **Save पर क्लिक करें।** जब आप अपनी सेटिंग्स के सही होने की पुष्टि कर चुके हों, तो **Save** बटन पर क्लिक कर दें।

- **आपकी नोटबुक का URL एड्रेस शेयर करें।** आखिर में, आपको अपने प्रोजेक्ट के URL एड्रेस को बस कॉपी-पेस्ट करके हमें भेजना है! **@girlswhocode #codefromhome** को टैग करना न भूलें।

### सहयोगियों के बारे में टिप्पणी: Kaggle

आपको आपकी नोटबुक में साथ मिलकर प्रोग्रामिंग करने के लिए अन्य उपयोक्ताओं को आमंत्रित करने की सुविधा देता है। यह सुविधा विभिन्न टीम के साथ मिलकर काम करने के लिए बहुत अच्छी है। **यू ही किसी को भी सहयोगी न बना लें!** इन संपादन अधिकारों को आप किनके साथ साझा करेंगे इस बारे में चयनशील रहें।

### परियोजना का लिंक



और Girls Who Code at Home परियोजनाओं के लिए बनी रहें!

