



Girls Who Code At Home

डेटा प्लेग्राउंड

Python में डेटा का जोड़-तोड़ (मेनिपुलेशन)

गतिविधि अवलोकन

क्या आप जानती हैं कि इंटरनेट द्वारा उत्पन्न लगभग 70% डेटा का उपयोग नहीं होता है? डेटा ऐसी कंपनियों के लिए अत्यंत उपयोगी हो सकता है जो - उदाहरण के लिए - डेटा का उपयोग करके आपके पसंदीदा फूड डिलीवरी एप में रेस्त्रां सुझाती हैं। पर डेटासेट्स सच में केवल उतने ही मूल्यवान होते हैं जितना उन्हें बनाने वाले लोगों की विविधता! यदि महिलाएं और अल्पसंख्यक वर्ग हमारा भविष्य तय करने वाली टेक्नॉलजी के विकास में शामिल न हो, तो हमारे जीवन को संचालित करने वाली प्रणालियों पर उन्हें बनाने वाले लोगों (अधिकांशतः श्वेत पुरुषों) के पूर्वाग्रहों का नियंत्रण होगा।

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! इस गतिविधि में आप सीखेंगे कि संभावित ट्रेंड्स का पता लगाने के लिए किसी डेटासेट से सूचनाएं कैसे निकालते हैं। **Python** का उपयोग करके हम सीखेंगे कि किसी डेटासेट विशेष को फ़िल्टर करके, उसमें संशोधन करके, उसमें से डेटा हटाकर और उस पर गणनाएं करके बुनियादी डेटा जोड़-तोड़ यानि बेसिक डेटा मेनिपुलेशन कैसे करते हैं। इस गतिविधि में हम विशिष्ट रूप से किकस्टार्टर ([Kickstarter](#)) प्रोजेक्ट्स से संबंधित डेटा पर नज़र डालेंगे; किकस्टार्टर एक क्राउडफंडिंग (बहुत से लोगों के योगदान से चलने वाला) प्लेटफ़ॉर्म है; और हम यह ज्ञात करेंगे कि किस श्रेणी के प्रोजेक्ट्स सबसे सफल हैं।

सीखने के लक्ष्य

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

- ◆ किसी समस्या के हल के लिए डेटा का उपयोग कैसे करें यह ज्ञात करने के लिए डीकम्पोज़ीशन (अपघटन, किसी बड़ी चीज़ को छोटे-छोटे टुकड़ों में विभक्त करना) का उपयोग करना।
- ◆ Python का उपयोग करके डेटा स्टोर करना, सर्च करना, मिटाना और संशोधित करना।
- ◆ यह समझना कि किसी डेटासेट में खोजबीन करने के लिए Kaggle और Jupyter Notebook का उपयोग कैसे करें।

सामग्री

- ◆ [Kaggle Kickstarter Data](#)
- ◆ [नमूना परियोजना](#)

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 हैं 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 रिव्रेश चाहिए हो तो हमारा सुझाव है कि आप हमारी यह गतिविधि देखें [क्या मैं आपकी मदद कर सकती हूँ?](#)

वुमन इन टेक स्पोर्टलाइट: थेरेसा जॉनसन



तस्वीर स्रोत: मध्यम

एक रिज़्यूमे एक दस्तावेज होता है जो एक व्यक्ति के कौशल और उपलब्धियों को प्रस्तुत करता है। वे आपके कौशल के बारे में गर्व करने और किसी भी भविष्य के नियोजता को यह बताने का अवसर देते हैं कि आप कितने अद्भुत व्यक्ति हैं! लेकिन, क्या आपने कभी **असफलता के रिज़्यूमे** के बारे में सुना है? अपने नियमित रिज़्यूमे को बनाए रखने और अपडेट करने के लिए, थेरेसा जॉनसन एक अलग रिज़्यूमे में अपनी विफलताओं के सभी दस्तावेज बनाती हैं। वह मानती है कि गलतियाँ बहुमूल्य अनुभव हैं जिनसे वह सीख सकती हैं! आप असफलता के रिज़्यूमे के बारे में **यहाँ** अधिक जान सकते हैं और यह कि उन्हें यह विचार **टीना सेलीग** स्टैनफोर्ड से कैसे मिला।

डॉ. थेरेसा जॉनसन **Airbnb** में एक उत्पाद डिजाइनर हैं, जहां उन्होंने एक डेटा वैज्ञानिक के रूप में भी काम किया है। Airbnb में काम करने से पहले उन्होंने स्टैनफोर्ड यूनिवर्सिटी से एयरोनॉटिक्स और एस्ट्रोनॉटिक्स में पीएचडी प्राप्त की। आप सोच रही होंगी कि ये सभी भूमिकाएँ और क्षेत्र आपस में कैसे संबंधित हैं? हो सकता है कि आप शुरू में Airbnb में डेटा से एयरोनॉटिक्स को न जोड़ें, लेकिन हम इसके मूल में डेटा का उपयोग और हेरफेर करते हैं। एक डेटा वैज्ञानिक के रूप में अपने समय के दौरान, थेरेसा ने विभिन्न बाजारों में घरों के लिए समीक्षा का अनुकूलन करने में मदद करने के लिए मैट्रिक्स का विश्लेषण करने पर काम किया। अब एक उत्पाद प्रबंधक के रूप में, वह भुगतान डेटा को समझने और कई टीमों में इस जानकारी को समन्वय और संचार करने के लिए मशीन लर्निंग और एआई के साथ अपने अनुभव का उपयोग करती हैं।

टेक उद्योग में रंगदार महिला के रूप में, थेरेसा उद्योग में विविधता के महत्व की लिए खुलके बोलती हैं। थेरेसा **StreetCode Academy** के बोर्ड के अध्यक्ष के रूप में भी कार्य करती हैं, जो सिलिकॉन वैली में रंगदारों के समुदायों के लिए मुफ्त प्रौद्योगिकी संसाधनों की आपूर्ति करने वाला जएक गैर-लाभकारी संगठन है। वे छात्राओं को कोडिंग, उद्यमशीलता और डिजाइन में लैपटॉप और कक्षाओं से जोड़ते हैं।

Airbnb में थेरेसा और उनके काम के बारे में अधिक जानने के लिए नीचे इस **वीडियो** को देखें।

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 थेरेसा 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।



तन्त्रकता

थेरेसा ने अपनी विफलताओं के रिज़्यूमे में विफलता के महत्व पर प्रकाश डाला है। उस समय के बारे में सोचें जब आपको असफलता का सामना करना पड़ा था और आप निराश हुए थे। आप अधिक सकारात्मक होने के लिए इस अनुभव को पुनः कैसे लिख सकते हैं? वह क्या चीज़ थी जो आपने अपने बारे में सीखी थी जिसे आप अपनी अगली चुनौती पर लागू कर सकते हैं?

परिवार के किसी सदस्य या मित्र के साथ अपनी प्रतिक्रियाएँ साझा करें। चर्चा में शामिल होने हेतु दूसरों को थेरेसा 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 हम चर्चा करेंगे कि बिग डेटा क्या होता है और हर उद्योग की कंपनियों के लिए यह मूल्यवान क्यों है।

बिग डेटा (2 मिनट)

बिग डेटा का अर्थ है बिग डेटा, यानी बहुत सारा डेटा! हम बिग डेटा और केवल 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, और प्रायः उसे “साफ” करने या उपयोग और व्याख्या के लायक तैयार बनाने के लिए बहुत प्रयास करने पड़ते हैं। 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 वर्णों 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।

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 दाहिने 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 (नोटबुक्स), 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 में जोड़-तोड़ (मेनिपुलेशन) शुरू करें, हमें समय निकाल कर उस डेटासेट को ठीक से जानना होगा जिसे हमने इस गतिविधि के लिए चुना है।

Kickstarter प्रोजेक्ट्स (1 मिनट)

Kickstarter एक क्राउडफंडिंग (बहुत से लोगों के योगदान से चलने वाला) प्लेटफॉर्म है जहां छोटे-छोटे व्यवसाय वेबसाइट पर कोई प्रोजेक्ट शुरू करके ऑनलाइन समुदाय से अपने प्रोजेक्ट के लिए छोटी-छोटी निवेश राशियां एकत्र 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? इस गतिविधि में हम आपको उन कुछ चीजों के बुनियादी उदाहरण समझाएंगे जो आप डेटा में जोड़-तोड़ यानि मेनिपुलेशन के लिए कर सकती हैं। Kickstarter डेटा में दो तरह की वैल्यूज (मान) होती हैं, पहली तो है [श्रेणीगत \(कैटेगोरिकल\)](#), यानि ऐसा डेटा जिसे श्रेणियों में समूहबद्ध किया जा सकता है, और दूसरी है संख्यात्मक (न्यूमेरिकल) जो आपको इस बात का सारांश दिखाती है कि अन्य डेटासेट्स को किस प्रकार निरूपित किया जा सकता है।

डेटासेट को छोटे-छोटे भागों में तोड़ना (5-8 मिनट)

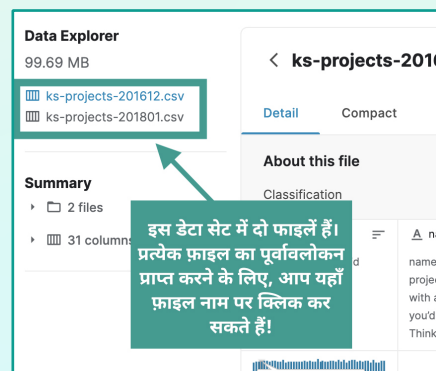


आइए, हमारे डेटा में गोता लगाएं! सबसे पहले, चलिए [Kaggle Kickstarter डेटासेट](#) को दोबारा खोलते हैं। सुनिश्चित करें कि आप **Data** टैब पर हों, आपको ध्यान देना चाहिए कि सबसे ऊपर मौजूद हेडर बार पर “Data” शब्द [नीला](#) है और [अंडरलाइन](#) किया हुआ है। अब नीचे स्क्रॉल करते हुए **Data Explorer** वाले भाग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

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।

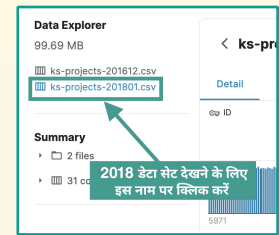
चूंकि हम सबसे हाल के डेटा के साथ कार्य करना चाहते हैं, अतः हम केवल **ks-projects-201801.csv** के साथ कार्य करेंगे, संक्षेप में हम इसे 2018 डेटासेट कहेंगे।



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!

चरण 2: डेटासेट में खोजबीन करें (जारी)

- ◆ **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 साइड 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! 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 **रो (पंक्ति)** एक अकेले Kickstarter प्रोजेक्ट को, या एक **एंटीटी को दर्शाती है**। उदाहरण के लिए, यदि हमारे पास मनुष्यों का कोई डेटासेट होता, तो उस डेटासेट में व्यक्ति एक **एंटीटी** 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 **एंटीटी**, रेश्मा सौजानी का और उसे परिभाषित करने वाले कुछ **फ़ीचर्स** का उदाहरण।

Kickstarter प्रोजेक्ट फ़ीचर्स (1 मिनट)

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 गौर 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 **दूढ़ने** के लिए मेन्यू में नीचे **स्कॉल** करना होगा जो डिफ़ॉल्ट दृश्य में प्रदर्शित नहीं हो रहे हैं।



समस्या को डीकम्पोज़ करना (5-10 मिनट)

अब हम डेटासेट और हमें दी गई जानकारी को थोड़ा समझ चुके हैं, तो अब हम क्या करें? यहां से, डेटा साइंटिस्ट उन प्रश्नों पर विचार-मंथन करते हैं जिनके उत्तर वे इस डेटासेट के साथ पाना चाहते हैं। हमारे विचार के लिए कुछ संभावित प्रश्न इस प्रकार हो सकते हैं:



- ◆ क्या प्रोजेक्ट की लंबाई और उसके सफल होने न होने के बीच कोई संबंध है?
- ◆ क्या प्रोजेक्ट के बैकर्स (संकल्प लेने वाले लोगों) की संख्या और उसके सफल होने न होने के बीच कोई संबंध है?
- ◆ क्या हम प्रोजेक्ट के वर्तमान संकल्पों के आधार पर पता लगा सकते हैं कि वह सफल होगा या नहीं?

इस डेटासेट के साथ हम और भी बहुत से प्रश्नों के उत्तर दे सकते हैं। यदि आप यह देखना चाहते हैं कि अन्य लोगों ने इस डेटासेट के साथ क्या किया है, तो बेहिचक Kaggle पर इस डेटासेट के [नोटबुक्स](#) देखें। हम इस गतिविधि में इस प्रश्न का उत्तर खोजेंगे कि:

प्रोजेक्ट्स की कौनसी मुख्य श्रेणी

की सफलता का प्रतिशत सबसे अधिक है?

इस अगले भाग में हम आपको बताएंगे कि प्रश्नों की एक शृंखला का उत्तर देकर इस प्रश्न को छोटे-छोटे भागों में कैसे तोड़ें। बड़ी समस्याओं को छोटे-छोटे भागों में तोड़ने की इस प्रक्रिया को **प्रॉब्लम डीकम्पोज़ीशन(समस्या अपघटन)** कहते हैं। कंप्यूटर साइंटिस्ट अक्सर इस विधि का उपयोग करके समस्या को बेहतर ढंग से समझते हैं और छोटे-छोटे खंडों में यह पता लगाते हैं कि उसका हल कैसे किया जाए।

प्रश्न 1

हमारे प्रश्न का उत्तर पाने के लिए हमें कौन-कौन से फ़ीचर्स चाहिए होंगे?

प्रश्न 2

हमें सफलता के प्रतिशत की गणना कैसे करनी चाहिए?

प्रश्न 3

इस प्रश्न का उत्तर पाने के लिए हमें कौनसे कार्य करने होंगे?

अगले पन्ने पर हमारे समाधानों पर नज़र डालने से पहले, यहां बेहिचक **थोड़ा रुक कर** इस बारे में थोड़ा विचार-मंथन करें कि आप इन प्रश्नों के उत्तर कैसे देते।

चरण 2: डेटासेट में खोजबीन करें (जारी)

प्रश्न 1: हमारे प्रश्न का उत्तर पाने के लिए हमें कौन-कौन से फ़ीचर्स चाहिए होंगे?

डेटासेट में शामिल फ़ीचर्स: ID, name, category, main_category, currency, deadline, goal, launched, pledged, state, backers, country, usd pledged, usd_pledged_real, usd_goal_real.

चूंकि हमारा प्रश्न प्रोजेक्ट्स की मुख्य श्रेणी (मेन कैटेगरी) के आधार पर उनकी सफलता का प्रतिशत ट्रेंडने से संबंध रखता है, अतः हम `state` और `main_category` नामक फ़ीचर्स का उपयोग करना चाहेंगे।

- ◆ हम `main_category` का उपयोग करना चाहते हैं, न कि `category` का, क्योंकि `main_category` फ़ीचर में केवल एक तय संख्या में विकल्प हैं और यह फ़ीचर अधिक सामान्य है। चूंकि `category` अधिक विशिष्ट है, इसलिए हो सकता है कि हमें ऐसी श्रेणियां (कैटेगरी) मिलें जिनसे बहुत कम प्रोजेक्ट्स संबंधित हैं, इस कारण हमें अशुद्ध परिणाम मिलेंगे।
- ◆ हम `state` फ़ीचर का उपयोग इसलिए करते हैं क्योंकि यही वह एकमात्र फ़ीचर है जो हमें बताता है कि कोई प्रोजेक्ट सफल हुआ था या नहीं।

प्रश्न 2: हमें सफलता के प्रतिशत की गणना कैसे करनी चाहिए?

प्रतिशत की गणना करने के लिए हमें सफलताओं की संख्या की किसी चीज़ से तुलना करनी होगी। चूंकि हम प्रति मुख्य श्रेणी (मेन कैटेगरी) सफलता की गणना करना चाहते हैं, अतः हमें सफलता की दर की निम्नवत गणना करनी होगी:

$$\% \text{ of success for category}_{\text{film+video}} = \frac{\# \text{ of successful projects in category}_{\text{film+video}}}{\# \text{ of total projects in category}_{\text{film+video}}} \times 100\%$$

ऊपर वाले उदाहरण में हमने दिखाया है कि फ़िल्म एंड वीडियो मेन कैटेगरी के प्रोजेक्ट्स की सफलता के % की गणना कैसे करनी है। हम हमारे डेटासेट में मौजूद **प्रत्येक श्रेणी (कैटेगरी)** के लिए इस प्रतिशत की गणना करना चाहेंगे। हालांकि, प्रत्येक श्रेणी (कैटेगरी) में प्रोजेक्ट्स की संख्या अलग-अलग हो सकती है, पर समीकरण में कोई बदलाव नहीं होना चाहिए।

प्रश्न 3: इस प्रश्न का उत्तर पाने के लिए हमें कौनसे कार्य करने होंगे?

किसी समस्या को छोटे-छोटे टुकड़ों में तोड़ते समय आपको यह सुनिश्चित करना चाहिए कि आपके कार्य/चरण छोटे-छोटे हों और उन्हें संभाला जा सकता हो। अभी यदि आपको अपने कुछ उप-चरणों के हल का तरीका नहीं पता तो कोई बात नहीं, मुद्दे की बात यह है कि अपनी बड़ी समस्या को हल करने की शुरुआत करने के लिए आपके पास एक आरंभ बिंदु होना चाहिए। हमारे मार्गदर्शक प्रश्न का उत्तर पाने के लिए हम जिन छोटे चरणों को हल करना चाहते हैं उनमें से कुछ इस प्रकार हैं।



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

डेटा साइंटिस्ट डेटा की प्रोसेसिंग और विश्लेषण के लिए बहुत से टूल्स और भाषाओं का उपयोग करते हैं। [R](#), [स्ट्रुक्चर्ड क्वेरी लैंग्वेज \(या SQL\)](#), और [Python](#) प्रोग्रामिंग भाषाओं के कुछ उदाहरण हैं। इस गतिविधि में हम [Pandas](#) नामक एक Python लाइब्रेरी की मदद से हमारे डेटा में जोड़-तोड़ यानि मेनिपुलेशन करने के लिए Python प्रोग्रामिंग भाषा के उपयोग पर ध्यान केंद्रित करेंगे।

Python (2 मिनट)



Python एक टेक्स्ट आधारित प्रोग्रामिंग भाषा है, यानि 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, यानि यह सभी के द्वारा उपयोग और आवश्यकतानुसार संशोधन के लिए मुक्त रूप से उपलब्ध है। अपडेट्स कैसे स्वीकारी जाती हैं और भाषा पर लागू की जाती हैं इस बारे में कड़े दिशानिर्देश मौजूद हैं, पर मूल रूप से कहें 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 कमांड्स और चर राशीयों 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 इंझट 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।

नई नोटबुक बनाना (5-8 मिनट)

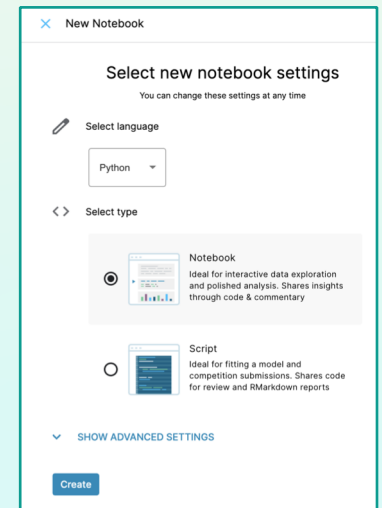
चलिए वापस हमारे [Kickstarter डेटासेट](#) पर लौटते हैं।

- **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। अपने डेटासेट को **मिनिमाइज़** करने के लिए, ऊपरी दाएं 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। आपके शीर्षक से आपके उस प्रश्न का थोड़ा-बहुत परिचय मिलना चाहिए जिसका आप उत्तर खोज रहे हैं। उदाहरण के लिए आप **Kickstarter** प्रोजेक्ट्स के लिए **Success Rate** और **Categories** का चयन कर सकते हैं। आप शीर्षक में अपना नाम शामिल करने का चयन भी कर सकते हैं, सुनिश्चित करें कि आप या तो केवल प्रथमाक्षर दिखाएं या अपना पहला नाम और अंतिम नाम का प्रथमाक्षर दिखाएं।

चरण 3: Python और Pandas का परिचय (जारी)

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">◆ import: यह कीवर्ड कंप्यूटर को बताता है कि हम एक Python लाइब्रेरी का उपयोग कर रहे हैं।◆ as: यह कीवर्ड, पैकेज को एक उपनाम देता है। यह एक वैकल्पिक चरण है पर इससे प्रोग्रामिंग आसान हो सकती है।◆ numpy/np: इस Python लाइब्रेरी का उपयोग डेटासेट पर गणितीय संक्रियाएं करने के लिए होता है। हमने इस पैकेज को np उपनाम दिया है।◆ pandas/pd: 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 pd उपनाम दिया है।

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, डेटासेट के अपने विश्लेषण को और बेहतर बनाने के लिए **numpy** लाइब्रेरी का उपयोग कैसे करते हैं।

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>	<p>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।</p>

चरण 3: Python और Pandas का परिचय (जारी)

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()` मेथड का उपयोग करेंगे। याद रखें कि **मेथड/फंक्शन** निर्देशों (कोड की पंक्तियाँ) का एक सेट होता है जो एक विशिष्ट कार्य करता है। आइए इस मेथड के सिन्टेक्स (वाक्य-रचना) को छोटे-छोटे टुकड़ों में समझें।

CSV, यानि 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
<code>pd.read_csv("filename")</code>	<ul style="list-style-type: none">◆ pd: यह कीवर्ड कंप्यूटर को बताता है कि हम <code>pandas</code> लाइब्रेरी में मौजूद एक मेथड का उपयोग कर रहे हैं। हम <code>pandas</code> की बजाय <code>pd</code> का उपयोग करते हैं क्योंकि हमने हमारे प्रोग्राम की शुरुआत में लाइब्रेरी इम्पोर्ट करते समय लाइब्रेरी को यही उपनाम दिया था।◆ .: यह सिम्बल कंप्यूटर को बताता है कि हम एक मेथड का उपयोग कर रहे हैं।◆ read_csv(): <code>pandas</code> में मौजूद यह मेथड, CSV फ़ाइल को पढ़ता है और उसे टेबल जैसे फ़ॉर्मेट में बदल देता है, ताकि Python में उसका उपयोग आसान हो जाए।◆ "filename": 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।

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।

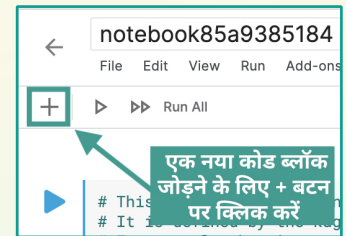
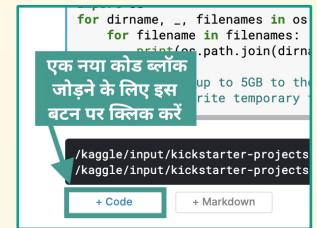
नया कोड ब्लॉक दो प्रकार से जोड़ सकते हैं:

- ◆ 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 Adelaide & Abullah	Poetry	Publishing	GBP
	1	1000003930	Greeting From Earth: ZGAC Arts Capsule For ET	Narrative Film	Film & Video	USD
	2	1000004038	Where is Hank?	Narrative Film	Film & Video	USD
	3	1000007540	ToshiCapital Rekordz Needs Help to Complete Album	Music	Music	USD
	4	1000011046	Community Film Project: The Art of Neighborhoods...	Film & Video	Film & Video	USD

	378656	999975400	ChienTruk Nationwide Charity Drive 2014 (Cancelled)	Documentary	Film & Video	USD
	378657	999977540	The Tribe	Narrative Film	Film & Video	USD
	378658	999986353	Walls of Remedy: New Indian Romantic Comedy 1...	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						

डीबगिंग 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 (फ़ुल 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 क्लोज़िंग 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">◆ <code>ds</code>: हम <code>info()</code> मेथड का उपयोग हमारे चर राशि <code>ds</code> पर करते हैं, न कि पूरी <code>pandas</code> लाइब्रेरी पर। ऐसा इसलिए है क्योंकि हम हमारे विशिष्ट डेटासेट के बारे में जानकारी पाना चाहते हैं।◆ <code>info()</code>: <code>pandas</code> में मौजूद यह मेथड, डेटासेट की जानकारी लेकर आती है, जिसमें फ़ीचर्स, रो (पंक्तियाँ), और डेटा टाइप शामिल हैं।

- 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

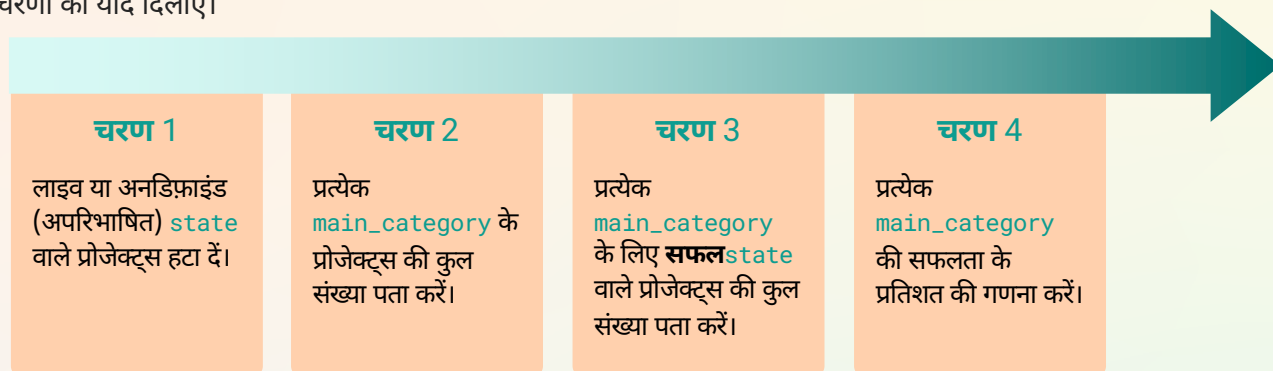
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 378661 entries, 0 to 378660
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   ID                    378661 non-null  int64  
 1   name                  378657 non-null  object  
 2   category              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 (फुल स्टॉप) शामिल किया हो।

चरण 3: डेटा में संशोधन करना (25-35 मिनट)

हमारे डेटासेट पर सफलता की दर की गणना शुरू करने से पहले, हमें यह सुनिश्चित करना होगा कि हमारे डेटा को क्लीन कर लिया गया हो। डेटा साइंटिस्ट्स को सबसे पहले ऐसे वैल्यूज़ को जांचना होता है जिनसे उनके विश्लेषण में त्रुटियां हो सकती हैं, जैसे डुप्लिकेट वैल्यूज़, स्पेलिंग की त्रुटियां, या खाली वैल्यूज़। इस प्रक्रिया को **डेटा क्लीनिंग** कहते हैं। इसमें गहरे उतरने से पहले, आइए हम खुद को हमारे उप-चरणों की याद दिलाएं।

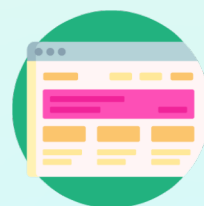


आपने देखा होगा कि जब आपने Kaggle पर Kickstarter डेटासेट पहली बार खोला था, तो वहां एक **यूज़ेबिलिटी (उपयोज्यता)** स्कोर दिया होता है। यूज़ेबिलिटी स्कोर से हमें पता चलता है कि डेटा कितना *क्लीन* है। इस डेटासेट का यूज़ेबिलिटी स्कोर 7.9 है जो काफ़ी हाई है! पर हमारे डेटा के लिए सफलता के प्रतिशत की गणना शुरू करने से पहले हमें अभी-भी हमारे डेटासेट की अतिरिक्त क्लीनिंग करने की ज़रूरत है।

याद करें कि इस डेटासेट में वे प्रोजेक्ट्स हैं जिनका `state` `success` (सफल), `cancelled` (रद्द), `undefined` (अपरिभाषित), `suspended` (स्थगित), `failed` (विफल), या `live` (लाइव) में से एक है। प्रोजेक्ट्स की सफलता दर की गणना करने में हम उन प्रोजेक्ट्स पर फ़ोकस करना चाहते हैं जो पहले ही पूरे हो चुके हैं। इसका यह अर्थ है कि हम ऐसे प्रोजेक्ट्स को शामिल नहीं करना चाहते जो **live (लाइव)** या **undefined (अपरिभाषित)** हैं, क्योंकि हम उनका अंतिम परिणाम अभी नहीं जानते हैं। इसे कैसे करना है इस बारे में गहरे जाने से पहले, आइए `pandas` में हमारे डेटासेट की संरचना के बारे में थोड़ा और जानते हैं।

Pandas DataFrame (10-15 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। यह काफ़ी हद तक जावास्क्रिप्ट के `Arrays` और Python के `Lists` जैसा है।



चूंकि हमारे डेटासेट में कुल **15 फ़ीचर्स**, या कॉलम्स हैं, तो हो सकता है हम यह सारा डेटा न देखना चाहें। चूंकि हम केवल दो फ़ीचर्स, `main_category` और `state` पर फ़ोकस करेंगे, आइए हमारे डेटासेट को केवल यही जानकारी दिखाने के लिए फ़ोकस करते हैं। आइए इन कॉलम्स को *चुनने (सलेक्ट करने)* और दर्शाने के कुछ घटकों का निर्माण करते हैं।

- **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

चरण 4: डेटा में संशोधन करना (जारी)

- **वांछित फ़ीचर्स के नामों वाली एक List बनाएं।** [] सिम्बल्स का उपयोग करके, हम स्क्वेयर ब्रैकेट्स (बड़े कोष्ठक) के अंदर फ़ीचर्स के नाम, कॉमा द्वारा अलग करते हुए जोड़ेंगे।


Python में, **List** एक क्रमबद्ध डेटा स्ट्रक्चर होती है जिसमें सूचनाएं रहती हैं। लिस्ट में वैल्यूज को एक क्रमांक, या **इंडेक्स** निर्धारित किया जाता है, जिससे वैल्यूज को एक्सेस करना, मिटाना या बदलना आसान हो जाता है।

PYTHON	विवरण
<code>["main_category", "state"]</code>	<ul style="list-style-type: none"> ◆ <code>[]</code>: स्क्वेयर ब्रैकेट्स यह दिखाते हैं कि हम Python में एक List बना रहे हैं। ◆ <code>"main_category", "state"</code>: हम उन फ़ीचर्स के नाम शामिल करते हैं जिनमें हमारी रुचि है। चूंकि ये नाम हैं, अतः हम प्रत्येक फ़ीचर के नाम को कोटेशन मार्क्स के अंदर लिखते हैं और वैल्यूज को कॉमा से अलग करते हैं।

- **हमारे डेटासेट चर राशि ds का उपयोग करके, फ़ीचर नेम्स की लिस्ट को कॉल करें।** 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 फ़ीचर्स की हमारी List को अंदर शामिल करना भी चाहते हैं।

PYTHON	विवरण
<code>ds[["main_category", "state"]]</code>	<ul style="list-style-type: none"> ◆ <code>ds[]</code>: हम चर राशि ds को कॉल करते हैं जिसमें हमारे डेटासेट का रेफ़रेंस मौजूद है, और [] सिम्बल्स का उपयोग करके यह दिखाते हैं कि हम हमारे डेटासेट में मौजूद सूचनाओं को एक्सेस करना चाहते हैं। ◆ <code>["main_category", "state"]</code>: हम फ़ीचर्स की यह List शामिल करके कंप्यूटर को यह बताते हैं कि हम हमारे डेटासेट से कौन-कौन से कॉलम्स चाहते हैं। यह जानकारी उन स्क्वेयर ब्रैकेट्स के अंदर जाती है जो हमारे डेटासेट को स्टोर करने वाले चर राशि ds के बाद आते हैं।

- **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
	<ul style="list-style-type: none"> ◆ 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। ◆ जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो। ◆ जांचें कि आपने फ़ीचर्स के नामों की स्पेलिंग सही लिखी हो। याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है। ◆ जांचें कि आपने अतिरिक्त स्क्वेयर ब्रैकेट्स [] न टाइप कर दिए हों। 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।

चरण 4: डेटा में संशोधन करना (जारी)

live (लाइव) या undefined (अपरिभाषित) प्रोजेक्ट्स हटाना (15-20 मिनट)

हम जो प्रोजेक्ट्स हमारे विश्लेषण में शामिल नहीं करना चाहते उन्हें हटाने के लिए, हमें Python में **कंडीशनल्स (conditionals)** का उपयोग करके कंप्यूटर को बताना होगा कि हमें कौनसे डेटा वैल्यूज़ चाहिए! हमारे विश्लेषण के लिए हमारे डेटासेट को तैयार करने के लिए, हम उन प्रोजेक्ट्स को हटाना चाहते हैं जो अभी पूरे नहीं हुए हैं। इसका अर्थ ऐसे प्रोजेक्ट्स हटाने से है जिनका **state** या तो **undefined (अपरिभाषित)** है या **live (लाइव)**। इन प्रोजेक्ट्स को कैसे हटाना है यह जानने से पहले, आइए हर स्टेट वाले प्रोजेक्ट्स की संख्याओं को देखे लेते हैं। इसे करने के लिए हम **value_counts()** नामक मेथड प्रयोग करेंगे।

→ **value_counts()** मेथड का उपयोग **state** फ़ीचर पर करें।

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

→ **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
असफल	197719	<ul style="list-style-type: none"> ◆ 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। ◆ जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो। ◆ 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। ◆ जांचें कि आपने अतिरिक्त स्क्वेयर ब्रैकेट्स [] न टाइप कर दिए हों।
सफल	133956	
निरस्त	38779	
अपरिभाषित	3562	
लाइव	2799	
निलंबित	1846	

चरण 4: डेटा में संशोधन करना (जारी)

जब हम हमारे डेटासेट से **live (लाइव)** और **undefined (अपरिभाषित)** प्रोजेक्ट्स को हटाने की कोशिश कर रहे हैं, तो ऐसे में हम प्रोजेक्ट्स की स्टेट-वार संख्याओं की तुलना इन ओरिजिनल वैल्यूज़ से कर सकते हैं। हम जो डेटा शामिल नहीं करना चाहते उसे फ़िल्टर करने के लिए, हमें कंडीशनल्स के एक संयोजन का उपयोग करना होगा। याद करें कि **सशर्त कथन (conditional statements)** तब कोड चलाते हैं **जब (if)** कोई शर्त, या नियमों का कोई समूह, संतुष्ट होता है। थोड़ा रुक कर यह सोचें कि हम जो चाहते हैं उसे एक **true** कंडीशनल स्टेटमेंट (सशर्त कथन) में कैसे बदलें। आइए एक-एक कदम करके देखते हैं कि हम हमारे नियम को Python के कंडीशनल स्टेटमेंट के रूप में कैसे लिखेंगे।

उदाहरण

नियम	Python कंडीशन (शर्त)
उदा.: सफल स्टेट (अवस्था) वाले प्रोजेक्ट्स	<code>ds["state"] == "successful"</code>

- `ds["state"]`: हम हमारे डेटासेट को फ़िल्टर करना चाहते हैं, हम हमारे डेटासेट चर राशि `ds` का उपयोग करते हैं, और फिर स्क्वेयर ब्रैकेट्स `[]` का उपयोग करके `"state"` फ़ीचर पर फ़िल्टरिंग निर्दिष्ट करते हैं।
- `== "successful"`: चूंकि हम केवल सफल प्रोजेक्ट्स चाहते हैं, अतः हम `==` सिम्बल का उपयोग करते हैं जो यह खोजता है कि कब स्टेट (state), "successful" (सफल) के बराबर है।

थोड़ा रुकें और बाकी के नियमों को Python के कंडीशनल स्टेटमेंट्स के रूप में नीचे टेबल में लिखने की कोशिश करें। कंडीशनल स्टेटमेंट्स (सशर्त कथनों) और/या कंपेरिज़न ऑपरेटर्स (तुलनाकारी संक्रियक) के बारे में रिव्रेशर चाहिए? Python में कंडीशनल्स के बारे में [W3 School का यह ट्यूटोरियल](#) देखें। चूंकि हम हमारे डेटासेट को फ़िल्टर करना चाहते हैं, हम हमारे डेटासेट चर राशि `ds` का उपयोग करते हैं, और फिर स्क्वेयर ब्रैकेट्स `[]` का उपयोग करके `"state"` फ़ीचर पर फ़िल्टरिंग निर्दिष्ट करते हैं।

नियम	Python कंडीशन (शर्त)
ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "undefined" (अपरिभाषित) नहीं है	<code>ds["state"]</code>
ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "live" (लाइव) नहीं है	<code>ds["state"]</code>



अगले पन्ने पर हमारे समाधान देखने से पहले यहां **थोड़ा रुकें**। हम जो नियम चाहते हैं उन्हें दर्शाने के लिए आप कई तरीकों से कंडीशनल स्टेटमेंट्स लिख सकते हैं। हमारे समाधान में बस एक तरीका बताया गया है, पर इसके कई समाधान हो सकते हैं।

नियम #1: ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "undefined" (अपरिभाषित) नहीं है

नियम	Python कंडीशन (शर्त)
ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "undefined" (अपरिभाषित) नहीं है	<code>ds["state"] != "undefined"</code>

चूंकि हम ऐसे प्रोजेक्ट्स चाहते हैं जिनका स्टेट (state), "undefined" (अपरिभाषित) नहीं है, अतः हम **!=** ऑपरेटर का उपयोग करते हैं। वैकल्पिक रूप से, आप एक ऐसा कंडीशनल स्टेटमेंट लिख सकते थे जो जांचता हो कि प्रोजेक्ट्स का स्टेट (state), failed (विफल), successful (सफल), cancelled (रद्द), या suspended (स्थगित) में से किसी एक के बराबर हो।

नियम #2: ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "live" (लाइव) नहीं है

नियम	Python कंडीशन (शर्त)
ऐसे प्रोजेक्ट्स जिनका स्टेट (state), "undefined" (अपरिभाषित) नहीं है	<code>ds["state"] != "undefined"</code>

उदाहरण की भांति, हमने **!=** ऑपरेटर का उपयोग ऐसे प्रोजेक्ट्स ढूंढने के लिए किया है जिनका स्टेट (state), "live" (लाइव) नहीं है। अब जबकि हम जान गए हैं कि हमें कौनसे कंडीशनल्स शामिल करने हैं, तो चलिए इसे हमारी नोटबुक में प्रोग्राम किए देते हैं।

- 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।
- ऐसे प्रोजेक्ट्स हटाने की शर्त (कंडीशन) लिखें जिनका `state="undefined"` (अपरिभाषित) और `"live"` (लाइव) हो। हम ऐसे प्रोजेक्ट्स को फ़िल्टर करने वाला कंडीशनल पहले ही देख चुके हैं जिनका `state` "undefined" (अपरिभाषित) और `"live"` (लाइव) नहीं है, पर इन नियमों को आपस में जोड़कर एक कंडीशनल स्टेटमेंट बनाने के लिए हमें **एंड (और)** ऑपरेटर का उपयोग करना होगा। हम pandas में एंड (और) को दिखाने के लिए **&** सिम्बल का उपयोग करते हैं।

pandas में **एंड** ऑपरेटर, Python के सामान्य कंडीशनल स्टेटमेंट की तुलना में अलग ढंग से दिखाया जाता है। चूंकि हम हमारे DataFrame में एक एंड (और) ऑपरेटर का उपयोग कर रहे हैं, इसलिए हम **&** सिम्बल का उपयोग करते हैं। यदि हम DataFrame से अलग कोई कंडीशनल स्टेटमेंट लिख रहे होते, तो हमने कीवर्ड **and** का उपयोग किया होता।

```
(ds["state"] != "undefined") & (ds["state"] != "live")
```

- ◆ **&:** यह कीवर्ड "and" को दर्शाता है और दो शर्तों को जोड़कर एक करने में हमारी मदद करता है।
- ◆ **():** हम जिन दो शर्तों (कंडीशन) के लिए जांच करना चाहते हैं उनके बीच भेद करने के लिए हम प्रत्येक कंडीशनल को कोष्ठकों के अंदर रखते हैं।

चरण 4: डेटा में संशोधन करना (जारी)

- **कंडीशनल स्टेटमेंट लागू करके डेटासेट फ़िल्टर करें।** यहां हम पिछले चरण वाले कंडीशनल स्टेटमेंट्स लागू करने के लिए `ds` चर राशि का और `[]` सिम्बल्स का उपयोग करेंगे। याद रखें कि कंडीशनल स्टेटमेंट्स, स्क्वेयर ब्रैकेट्स `[]` के अंदर होने चाहिए।

```
ds[(ds["state"] != "undefined") & (ds["state"] != "live")]
```


- **इस नए-नए फ़िल्टर हुए डेटासेट को `compProj` नामक 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। हमने हमारे चर राशि को `compProj` नाम दिया है, जो कम्प्लीटेड प्रोजेक्ट्स का संक्षिप्तीकरण है।

```
compProj = ds[(ds["state"] != "undefined") & (ds["state"] != "live")]
```

- **`value_counts()` मेथड का उपयोग करके हमारे फ़िल्टर किए हुए डेटा में प्रोजेक्ट्स की स्टेट-वार संख्याएं देखें।**

```
compProj["state"].value_counts()
```

- **अपना कोड ब्लॉक चलाएं।** अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें। आपको निम्नलिखित परिणाम मिलने चाहिए:

परिणाम		डीबगिंग के सुझाव
असफल	197719	<ul style="list-style-type: none">◆ 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।◆ जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो।◆ जांचें कि आपने फ़ीचर्स के नामों की स्पेलिंग सही लिखी हो। याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।◆ जांचें कि आपने अतिरिक्त स्क्वेयर ब्रैकेट्स <code>[]</code> न टाइप कर दिए हों।◆ जांचें कि हर कंडीशनल, कोष्ठकों <code>()</code> के अंदर लिखा हो।
सफल	133956	
निरस्त	38779	
निलंबित	1846	

ध्यान दें कि हमारे फ़िल्टर किए हुए डेटासेट में अब एक भी ऐसा प्रोजेक्ट नहीं है जो "undefined" (अपरिभाषित) या "live" (लाइव) के रूप में सूचीबद्ध हो। आपको इस बात पर भी ध्यान देना चाहिए कि "failed" (विफल), "successful" (सफल), "canceled" (रद्द), और "suspended" (स्थगित) स्टेट वाले प्रोजेक्ट्स की संख्याओं में कोई बदलाव नहीं हुआ है। यदि आपको कोई त्रुटि मिलती है, तो निम्नलिखित चीज़ें जांचें

ओहो, हमने तो बहुत कुछ पूरा कर लिया! अब जबकि हम हमारे डेटा को क्लीन कर चुके हैं, तो चलिए हमारी विश्लेषक वाली टोपी पहनते हैं और सफलता की दर की गणना शुरू करते हैं!

चरण 5: सफलता के प्रतिशत की गणना करना (15-20 मिनट)

अब जबकि हमारा डेटा क्लीन हो गया है, हम सफलता के प्रतिशत की गणना करना शुरू कर सकते हैं। चलिए हम खुद को हमारे उप-चरणों की याद दिलाएं।



आप मानें या न मानें, आपको पहले ही पता है कि बाकी के चरणों की कोडिंग कैसे करनी है! हम प्रत्येक केटेगरी में प्रोजेक्ट्स की संख्या जानने के लिए `value_counts()` मेथड का और केवल सफल प्रोजेक्ट्स हेतु फ़िल्टर करने के लिए **कंडीशनल्स** का उपयोग करेंगे। चिंता न करें, हम आपको सभी चरणों के बारे में विस्तार से बताएंगे!

याद रखें, सफलता के प्रतिशत की गणना करने के लिए हमें प्रत्येक केटेगरी के सफल प्रोजेक्ट्स की संख्या की तुलना प्रोजेक्ट्स की कुल संख्या से करनी होगी।

$$\% \text{ of success for category}_{\text{film+video}} = \frac{\# \text{ of successful projects in category}_{\text{film+video}}}{\# \text{ of total projects in category}_{\text{film+video}}} \times 100\%$$

ऊपर वाले उदाहरण में हमने दिखाया है कि फ़िल्म एंड वीडियो मेन केटेगरी के प्रोजेक्ट्स की सफलता के % की गणना कैसे करनी है। हम हमारे डेटासेट में मौजूद **प्रत्येक श्रेणी (केटेगरी)** के लिए इस प्रतिशत की गणना करना चाहेंगे। हालांकि, प्रत्येक श्रेणी (केटेगरी) में प्रोजेक्ट्स की संख्या अलग-अलग हो सकती है, पर समीकरण में कोई बदलाव नहीं होना चाहिए।

सबसे पहले हम, प्रत्येक केटेगरी के सफल प्रोजेक्ट्स की संख्या ज्ञात करेंगे। उसके बाद हम प्रत्येक केटेगरी के प्रोजेक्ट्स की कुल संख्या ज्ञात करेंगे। आखिर में हम इन दो वैल्यूज़ का उपयोग करके सफलता के प्रतिशत की गणना करेंगे।

चरण 5: सफलता के प्रतिशत की गणना करना (जारी)

प्रति श्रेणी (केटेगरी) प्रोजेक्ट्स की कुल संख्या (3-5 मिनट)

- 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।
- प्रत्येक `main_category` के प्रोजेक्ट्स की कुल संख्या को `totProjCount` नामक चर राशि में स्टोर करें। हम हमारे फ़िल्टर किए हुए डेटासेट, `compProj` के `"main_category"` फ़ीचर पर `value_counts()` मेथड का उपयोग करेंगे। `value_counts()` मेथड एक **Series** ऑब्जेक्ट रिटर्न करता है, जो वैल्यूज़ की एक लिस्ट के जैसा होता है। इससे हमारे लिए बस एक लाइन के कोड से हर केटेगरी की सफलता के प्रतिशत की गणना करना आसान हो जाएगा। हमने हमारी चर राशि को `totProjCount` नाम दिया है, जो टोटल नंबर ऑफ़ प्रोजेक्ट्स का संक्षिप्तीकरण है।
- `totProjCount` को प्रिंट आउट करें और अपना कोड ब्लॉक चलाएं। यह एक वैकल्पिक चरण है, पर यह इस बात की जांच करने के लिए अच्छा है कि आगे बढ़ने से पहले हम कुल प्रोजेक्ट्स की सही संख्या प्राप्त कर पा रहे हों। `print()` मेथड का उपयोग करके `totProjCount` को प्रिंट आउट करें और फिर अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें।

पहले हमने प्रत्येक प्रकार के `state` के लिए प्रोजेक्ट्स की संख्या ज्ञात करने हेतु `value_counts()` मेथड का उपयोग किया था। यहां हम हमारे डेटासेट को श्रेणियों (केटेगरी) के अनुसार तोड़ना चाहते हैं, इसलिए हमने उसके स्थान पर `"main_category"` फ़ीचर जोड़ा है।

```
print(totProjCount)
```

परिणाम

फिल्म और वीडियो	62399
संगीत	49403
पब्लिशिंग	39113
गेम्ज	34943
प्रौद्योगिकी	32189
डिजाइन	29763
आर्ट	27959
भोजन	24418
फैशन	22563
थिएटर	10871
कॉमिक्स	10743
फोटोग्राफी	10730
क्राफ्ट्स	8733
पत्रकारिता	4724
नृत्य	3749

डीबगिंग के सुझाव

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत पड़ सकती है। अपनी नोटबुक में सबसे ऊपर दो तीरों वाले बटन `>>` पर क्लिक करें।
- ◆ जांच लें कि आप हमारे फ़िल्टर किए हुए डेटासेट, `compProj` का उपयोग कर रहे हों।
- ◆ जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो।
- ◆ जांच लें कि आपके नामों (फ़ीचर्स, टेबल और मेथड्स) के स्पेलिंग सही हों। याद रखें कि Python भी केस सेंसिटिव है।
- ◆ जांचें कि आपने अतिरिक्त स्क्वेयर ब्रैकेट्स `[]` न टाइप कर दिए हों।
- ◆ जांच लें कि आप `value_counts` मेथड के उपयोग के बाद कोष्ठक शामिल करें।

प्रति श्रेणी (कटेगरी) सफल प्रोजेक्ट्स की संख्या (3-5 मिनट)

- 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।
- केवल सफल प्रोजेक्ट्स वाले डेटासेट का एक नया रेफरेंस `susProj` नामक चर राशि में स्टोर करें। आइए `compProj` डेटासेट को फ़िल्टर करके एक ऐसा डेटासेट बनाएं जिसमें केवल "successful" (सफल) स्टेट वाले प्रोजेक्ट्स हों।

```
susProj = compProj[compProj["state"] == "successful"]
```

- प्रत्येक `main_category` के सफल प्रोजेक्ट्स की कुल संख्या को `susProjCount` नामक चर राशि में स्टोर करें। हम यह सुनिश्चित करना चाहते हैं कि हम केवल सफल प्रोजेक्ट्स वाले हमारे डेटासेट, `susProj` के "main_category" फ़ीचर पर `value_counts()` मेथड का उपयोग करें।

```
susProjCount = susProj["main_category"].value_counts()
```

- `susProjCount` को प्रिंट आउट करें और अपना कोड ब्लॉक चलाएं। यह एक वैकल्पिक चरण है, पर यह इस बात की जांच करने के लिए अच्छा है कि आगे बढ़ने से पहले हम कुल प्रोजेक्ट्स की सही संख्या प्राप्त कर पा रहे हों। `print()` मेथड का उपयोग करके `susProjCount` को प्रिंट आउट करें और फिर अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें।

```
print(susProjCount)
```

परिणाम

संगीत	24197
फिल्म और वीडियो	23623
गेम्ज़	12518
पब्लिशिंग	12300
आर्ट	11510
डिजाइन	10550
थिएटर	6534
प्रौद्योगिकी	6434
भोजन	6085
कॉमिक्स	5842
फैशन	5593
फोटोग्राफी	3305
नृत्य	2338
क्राफ्ट्स	2115
पत्रकारिता	1012

डीबगिंग के सुझाव

- ◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत पड़ सकती है। अपनी नोटबुक में सबसे ऊपर दो तीरों वाले बटन `⏮` पर क्लिक करें।
- ◆ जांच लें कि आप हमारे फ़िल्टर किए हुए डेटासेट, `susProj` का उपयोग कर रहे हों।
- ◆ जांचें कि आपने अपने फ़ीचर्स के नामों को कोटेशन मार्क्स के अंदर लिखा हो।
- ◆ जांच लें कि आपके नामों (फ़ीचर्स, टेबल और मेथड्स) के स्पेलिंग सही हों। याद रखें कि Python भी केस सेंसिटिव है यानि इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
- ◆ जांचें कि आपने अतिरिक्त स्क्वेयर ब्रैकेट्स `[]` न टाइप कर दिए हों।
- ◆ जांच लें कि आप `value_counts` मेथड के उपयोग के बाद कोष्ठक शामिल करें।

चरण 5: सफलता के प्रतिशत की गणना करना (जारी)

सफलता के प्रतिशत की गणना करना (3-5 मिनट)

- अपनी नोटबुक में सबसे नीचे एक नया कोड ब्लॉक जोड़ें।
- `susProjCount` और `totProjCount` चर राशि का उपयोग करके, सफलता के प्रतिशत की गणना करें। `pandas` की एक सबसे अच्छी विशेषता यह है कि आप बस एक पंक्ति के कोड से कई फ़ीचर्स पर समान गणना कर सकते हैं! हम प्रत्येक श्रेणी (केटेगरी) की सफलता के प्रतिशत की गणना `susProjCount` में `totProjCount` का भाग देने मात्र से कर सकते हैं। हमारा प्रतिशत मान पाने के लिए 100 से गुणा करना न भूलें।

$$\text{susProjCount} / \text{totProjCount} * 100$$

- अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें।

परिणाम		डीबगिंग के सुझाव
आर्ट	41.167424	<ul style="list-style-type: none">◆ आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत पड़ सकती है। अपनी नोटबुक में सबसे ऊपर दो तीरों वाले बटन <code>>></code> पर क्लिक करें।◆ जांचें कि आपने 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।◆ अपना उत्तर प्रतिशत रूप में पाने के लिए 100 से गुणा करना न भूलें।
कॉमिक्स	54.379596	
क्राफ्ट्स	24.218482	
नृत्य	62.363297	
डिजाइन	35.446696	
फैशन	24.788370	
फिल्म और वीडियो	37.857978	
भोजन	24.920141	
गेम्स	35.824056	
पत्रकारिता	21.422523	
संगीत	48.978807	
फोटोग्राफी	30.801491	
पब्लिशिंग	31.447345	
प्रौद्योगिकी	19.988195	
थिएटर	60.104866	

चरण 5: सफलता के प्रतिशत की गणना करना (जारी)

अपने परिणामों को क्रमबद्ध करना (3-5 मिनट)

आपने देखा होगा कि आपके अंतिम परिणाम कैटेगरी के नाम के अनुसार अंग्रेज़ी वर्णमाला के क्रम में व्यवस्थित हैं। आप अपने परिणामों को `sort_values()` मेथड का उपयोग करके क्रमबद्ध कर सकते हैं। आइए हमारे पिछले कोड ब्लॉक को इस प्रकार अपडेट करते हैं कि वह क्रमबद्ध वैल्यूज़ का आउटपुट दे।

→ सफलता के प्रतिशत को `percentSuccess` नामक चर राशि में स्टोर करें।

```
percentSuccess = susProjCount/totProjCount * 100
```

→ `sort_values()` मेथड का उपयोग करके प्रतिशतताओं को क्रमबद्ध करें।

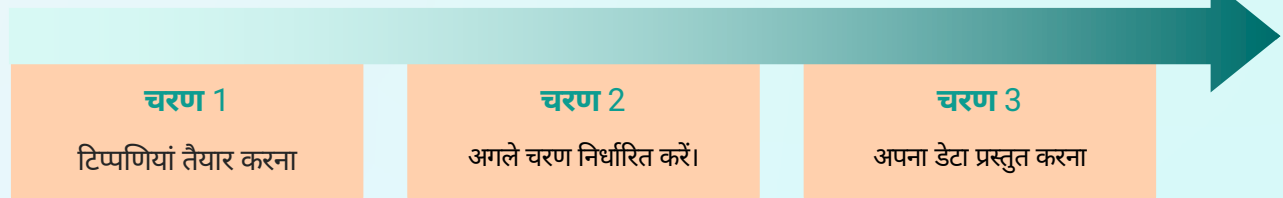
```
percentSuccess.sort_values()
```

→ अपना कोड ब्लॉक चलाएं। अपना कोड चलाने के लिए कोड ब्लॉक के बायीं ओर मौजूद नीला प्ले बटन क्लिक करें।

परिणाम	डीबगिंग के सुझाव
प्रौद्योगिकी 19.988195	<ul style="list-style-type: none">आपको अपने सारे कोड ब्लॉक्स दोबारा चलाने की ज़रूरत पड़ सकती है। अपनी नोटबुक में सबसे ऊपर दो तीरों वाले बटन ▶▶ पर क्लिक करें।जांचें कि आपने नामों की स्पेलिंग सही लिखी हो। याद रखें कि Python भी केस सेंसिटिव है यानी इसमें अपरकेस और लोअरकेस अक्षरों को समान नहीं माना जाता है।
पत्रकारिता 21.422523	
क्राफ्ट्स 24.218482	
फैशन 24.788370	
भोजन 24.920141	
फोटोग्राफी 30.801491	
पब्लिशिंग 31.447345	
डिजाइन 35.446696	
गेम्ज़ 35.824056	
फिल्म और वीडियो 37.857978	
आर्ट 41.167424	
संगीत 48.978807	
कॉमिक्स 54.379596	
थिएटर 60.104866	
नृत्य 62.363297	

चरण 5: अंतिम परिणामों पर विचार-मंथन (15-20 मिनट)

भविष्य के ट्रेंड्स के बारे में निष्कर्ष निकालते समय अपने विश्लेषण पर विस्तार से विचार-मंथन करना अत्यधिक महत्वपूर्ण होता है। यह उतना ही महत्वपूर्ण है जितना आपकी विश्लेषण प्रक्रिया के अन्य सभी चरण! विचार-मंथन से आप निम्नलिखित चरणों पर फ़ोकस करेंगे:



इस अनुभाग में हम आपको उस विचार-मंथन प्रक्रिया का एक सारांश देंगे जिसे डेटा साइंटिस्ट बिग डेटा पर अपना विश्लेषण पूरा करने के बाद संचालित करते हैं। इस गतिविधि में हम आपको प्रक्रिया के केवल पहले और दूसरे चरण के बारे में विस्तार से बताएंगे। अगले सप्ताह के लिए तैयार रहें जब हम आपको बताएंगे कि अपने डेटा को प्रस्तुत करने के लिए आप Python में मौजूद विभिन्न विज़ुअलाइज़ेशन तकनीकों का किस प्रकार उपयोग कर सकते हैं!

चरण 6: सफलता के प्रतिशत की गणना करना (जारी)

प्रेक्षण करना (6-10 मिनट)

3-5 मिनट निकाल कर अंतिम परिणामों पर नज़र डालें और अपने प्रेक्षणों को नोट करें। यहां हमारे सामने समय की कमी रूपी बाधा है, अतः आप उस डेटा पर फ़ोकस कर सकते हैं जो आपको सबसे असाधारण दिखे - यदि आप चाहें तो जब चाहें तब लौटकर इन्हें दोबारा देख सकते हैं! इस समय के दौरान, अपने डेटा के बारे में जिन भी प्रेक्षणों पर आपका ध्यान जाए उन्हें आपको नोट कर लेना चाहिए। केवल तथ्यों/डेटा पर फ़ोकस करें और कोई विचार कायम करने की कोशिश न करें।

अवलोकन <i>कौनसा डेटा आपको असाधारण दिखा? डेटा में मौजूद तथ्यों पर टिके रहें और फिलहाल कोई निष्कर्ष निकालने की कोशिश न करें।</i>	डेटा स्रोत <i>आपको यह जानकारी कहाँ मिली? डेटासेट का नाम और/या कोड की पंक्तियाँ शामिल करें।</i>
उदा. Film & Video केटेगरी में सबसे अधिक कुल 62,399 प्रोजेक्ट्स हैं।	उदा.: In totProjCount. "undefined" (अपरिभाषित) और "live" (लाइव) प्रोजेक्ट्स को हटाकर, फ़िल्टर किए हुए डेटासेट से योग की गणना करें।

थोड़ा रुकें और अपने टेबल में आपने जो प्रेक्षण लिखे थे उन पर गौर करें। इस अनुभाग में, हम हमारे इस मुख्य प्रश्न का उत्तर देंगे कि: **किस मुख्य श्रेणी (मेन केटेगरी) के प्रोजेक्ट्स की सफलता की दर सबसे अधिक है?**

अब, हमारे मुख्य प्रश्न का उत्तर देने के लिए, और आपको जो भी अन्य चीज़ रोचक लगी हो उसकी पहचान करने करने और यह बताने के लिए कि वह आपको रोचक क्यों लगी, **3 मिनट** का टाइमर सेट कर दें। आप चाहें तो इस बारे में विचार कर सकते हैं कि आपके विचार में आपका डेटा, सामान्य रूप से Kickstarter प्रोजेक्ट्स और उनकी श्रेणियों के बारे में क्या कहता है। बैकर (संकल्प करने वाले व्यक्ति) के नज़रिए से सोचें, यदि आपको किसी प्रोजेक्ट के लिए संकल्प करना होता, तो आपके विचार में किस श्रेणी के प्रोजेक्ट में सफलता की संभावना सबसे अधिक होती?

आगे के चरणों का निर्धारण (3 मिनट)

अब, जबकि आपने अपने डेटा के संबंध में कुछ प्रेक्षण कर लिए हैं, आपको आगे क्या करना है? इस अनुभाग में, थोड़ा समय निकाल कर उन बाधाओं के बारे में सोचें जो आपके विश्लेषण में मौजूद थीं, और ऐसी कुछ उन चीज़ों के बारे में सोचें जिनकी आप इस डेटासेट के साथ और खोजबीन करना चाहेंगे। आपके शोध में कौनसी अतिरिक्त जानकारी सहायक सिद्ध होगी? **3 मिनट** लेकर निम्नलिखित प्रश्नों पर विचार करें:

प्रश्न 1

अपने डेटा का विश्लेषण करने के दौरान आपका सामना किन चुनौतियों से हुआ था?

प्रश्न 2

डेटा के बारे में आप कौनसे अतिरिक्त प्रश्न जानना चाहेंगे?

प्रश्न 3

आपको अपना विश्लेषण और आगे ले जाने के लिए किस अतिरिक्त जानकारी की ज़रूरत होगी?

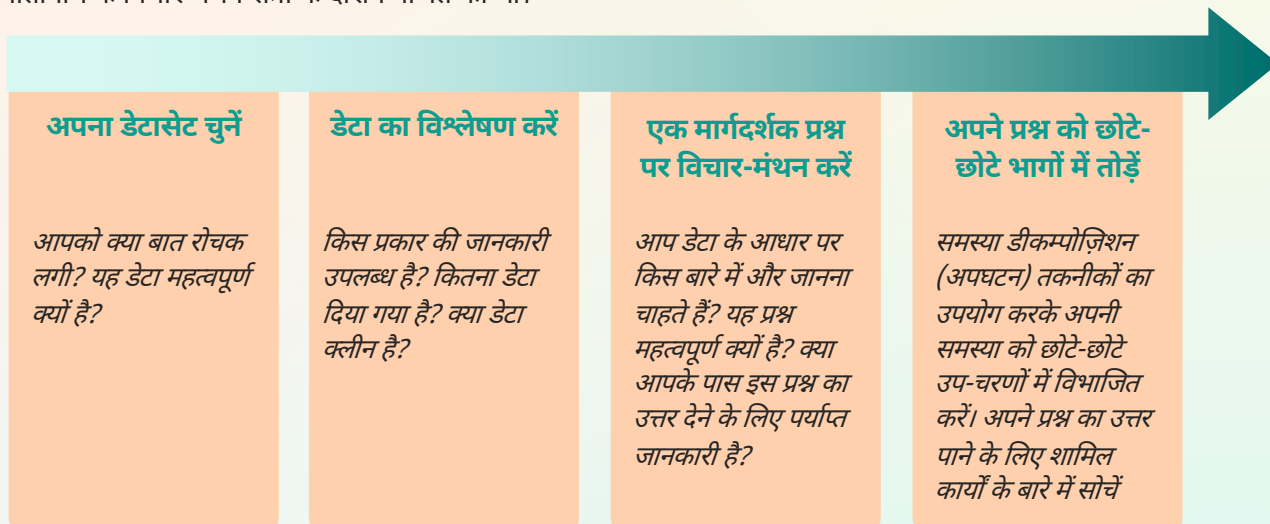
प्रायः डेटा के साथ कार्य करते समय आप अपने निष्कर्षों/जांच-परिणामों को विभिन्न प्रकार के विज़ुअलाइज़ेशन के साथ प्रस्तुत करना चाहेंगे, जिनमें चार्ट और ग्राफ शामिल हैं। इस गतिविधि में हमने जाना कि डेटा में जोड़-तोड़ यानि मेनिपुलेशन कैसे करें, पर हमारी अगली गतिविधि के लिए हमारे साथ बने रहिए जिसमें हम Python में मौजूद डेटा विज़ुअलाइज़ेशन की तकनीकों के बारे में जानेंगे। फिलहाल के लिए, एक ब्रेक लीजिए, और एक दिन का डेटा साइंटिस्ट बनने के लिए खुद की पीठ थपथपाएं!

चरण 7: 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 चाहें तो परिणामों को **यूज़ेबिलिटी (उपयोज्यता)** के अनुसार क्रमबद्ध कर सकते हैं, याद करें कि इस स्कोर से आपको यह पता करने में मदद मिलती है कि कोई डेटासेट कितना “क्लीन” है।

अपने चुने हुए डेटासेट पर अपना स्वयं का विश्लेषण करते समय, आप वैसी ही प्रक्रिया का पालन करना चाहते हैं जैसी हमने इस गतिविधि के विचार-मंथन सत्रों के दौरान वर्णित की थी।



जब आप अपना डेटासेट चुन चुके हों और डेटा का विश्लेषण कर सकने के तरीकों पर विचार-मंथन कर चुके हों, तो यह अपना शोध शुरू करने का समय होता है! कुछ शोधकार्यों में कई घंटे, दिन, यहां तक कि महीने भी लग सकते हैं। हौसला न खोएं। इस गतिविधि में हमने ऐसी कुछ चीज़ों को मात्र सतही तौर पर जाना है जिनसे [pandas](#) डेटा विश्लेषण 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!

- [DataCamp का Pandas ट्यूटोरियल: Python में डेटाफ्रेम्स \(DataFrames in Python\)](#)
- [LearnDataSci का Python Pandas ट्यूटोरियल](#)
- [Tutorials Point का Python Pandas ट्यूटोरियल](#)
- [DataCamp की Pandas चीट शीट](#)

एक्सटेंशन 2: विफलता का प्रतिशत ढूँढना (10-15 मिनट)

हमने सफलता का प्रतिशत तो ढूँढ लिया, पर यदि हम विफलता का प्रतिशत ढूँढना चाहते हों तो? हम इसे कई अलग-अलग तरीकों से कर सकते हैं।

- यदि हम विफलता को सफलता के *विलोम* मात्र के रूप में देखें और यह देखें कि अंतिम अवस्थाएं/स्टेट्स (states) केवल दो ही हैं, तो हम 100% में से सफलता का प्रतिशत घटाने मात्र से विफलता प्रतिशत ज्ञात कर सकते हैं।

```
percentFail = 1 - percentSuccess
```

याद रखें कि मूल रूप से हमने “undefined” (अपरिभाषित) और “live” (लाइव) स्टेट वाले प्रोजेक्ट्स फ़िल्टर कर दिए थे।

याद करें कि हमारे डेटासेट में कई अलग-अलग अवस्थाएं/स्टेट्स (states) थीं: successful (सफल), failed (विफल), canceled (रद्द), undefined (अपरिभाषित), live (लाइव), और suspended (स्थगित)। 100% से सफलता के प्रतिशत को घटाने मात्र के द्वारा हम यह मानकर चल रहे हैं कि हम “canceled” (रद्द) और “suspended” (स्थगित) स्टेट्स को “failed” (विफल) के समान मानते हैं।

- मान लें कि हम केवल “failed” (विफल) स्टेट वाले प्रोजेक्ट्स को गिनते हुए विफलता के प्रतिशत की गणना करना चाहते हैं। इसके लिए हमें और कुछ नहीं बस हमारे डेटासेट पर `value_counts()` मेथड का उपयोग करना है।

```
failProj = compProj[compProj["state"]=="failed"]
failProjCount = failProj["main_category"].value_counts()
percentFail = failProjCount/totProjCount * 100
```

एक्सटेंशन 3: अपने डेटासेट में वैल्यूज़ बदलना (5-10 मिनट)

कभी-कभी आप अपने डेटासेट में वैल्यूज़ को बदलना चाह सकते हैं। उदाहरण के लिए, आप कुछ श्रेणियों (केटेगरीज़) को संयुक्त करना चाह सकते हैं, या फिर आप यह सुनिश्चित करना चाह सकते हैं कि कुछ वैल्यूज़ को एकसमान पढ़ा जाए, जैसे “Film & Video” और “Film and Video” समान हैं। हम `pandas` में `replace()` मेथड का उपयोग करके यह कार्य बड़ी आसानी से कर सकते हैं। आप `replace` मेथड का उपयोग और किन-किन परिस्थितियों में कर सकते हैं यह जानने के लिए इस [संसाधन](#) पर एक नज़र डालें।

चरण 8: 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)

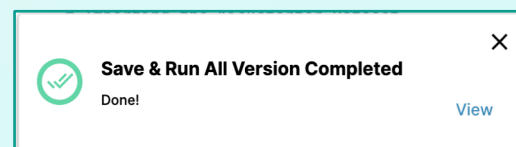
- **Save Version बटन पर क्लिक करें।** आपने अपनी नोटबुक के ऊपरी दायें कोने में **Save Version** बटन देखा होगा। इससे एक नई विंडो खुलनी चाहिए।
- **एक Version Name जोड़ें।** यह वैकल्पिक फ़ील्ड आपके लिए यह दर्ज करने का एक अच्छा तरीका है कि आपने इस नए वर्ज़न (संस्करण) में ऐसा क्या किया है जो पिछले संस्करणों 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 8: 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।



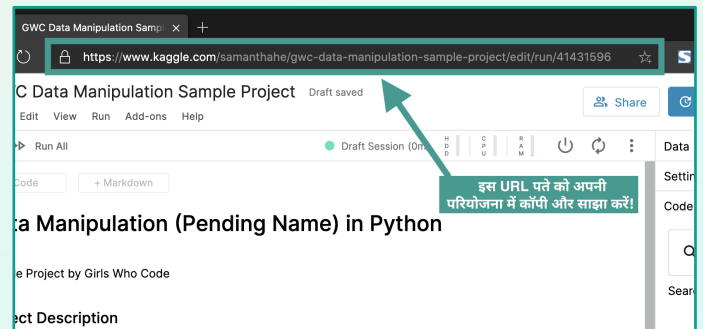
- **Privacy को बदलकर Public कर दें।** Privacy फ़ील्ड के दाहिनी ओर वाला ड्रॉप डाउन कॉलम चुनें और उसमें से **Public** चुनें। इससे एक चेतावनी संदेश दिखेगा कि सुनिश्चित करें कि आप इस बात से अवगत हैं कि अन्य उपयोक्ता आपका प्रोजेक्ट देख सकेंगे। **Ok, make public** चुनें।



- **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!

परियोजना का लिंक



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

