লক্ষ্য হলো বিভিন্ন উৎস থেকে ডেটা আনতে শেখা। চারটি প্রধান ডেটা আনার পদ্ধতির কথা বলা হয়েছে:

1. **CSV (Comma Separated Values)** ফাইলের সাথে কাজ করা**:** এটি সবচেয়ে সহজ এবং ML শেখার শুরুতে এটিই সবচেয়ে বেশি ব্যবহৃত ডেটা ফরম্যাট।
2. **JSON (JavaScript Object Notation)** এবং **SQL:** JSON হলো একটি বিশ্বব্যাপী স্বীকৃত ফরম্যাট, যা সাধারণত API-এর মাধ্যমে ডেটা আনার সময় ব্যবহার করা হয়। SQL ব্যবহার হয় ডেটাবেস থেকে ডেটা আনার জন্য।
3. **API** থেকে ডেটা আনা**:** কোনো ওয়েবসাইটের সার্ভার থেকে ডেটা নেওয়া এবং প্রয়োজন অনুযায়ী তা পরিবর্তন করা।
4. ওয়েব স্ক্র্যাপিং**:** যে ওয়েবসাইটে API নেই, সেখান থেকে ডেটা বের করে আনতে একটি পার্সার ব্যবহার করে ওয়েবসাইটের HTML কোড নেভিগেট করা।

এই চারটি পদ্ধতি শিখলে সাধারণত আপনার **90%** সমস্যা সমাধান হয়ে যাবে।

CSV ফাইল হ্যান্ডেল করার জন্য `read_csv` ফাংশন

CSV এবং TSV (Tab Separated Values) ফাইল লোড করার জন্য প্রধানত `read_csv` ফাংশনটি ব্যবহৃত হয়। CSV-তে ডেটা কমা দিয়ে আলাদা করা হয়, আর TSV-তে ট্যাব ব্যবহার করা হয়। বাস্তব ডেটা সেটের নানা সমস্যা সমাধানের জন্য এই ফাংশনে প্রচুর প্যারামিটার রয়েছে।

১. ডেটা লোড করা

- লোকাল এবং সার্ভার ফাইল**:** ফাইল আপনার কম্পিউটার বা লোকাল মেশিনে থাকুক, অথবা কোনো সার্ভারের URL-এ থাকুক, আপনি এটি লোড করতে পারেন। সার্ভার থেকে ডেটা আনার জন্য `requests` লাইব্রেরি ব্যবহার করে ফাইলের কন্টেন্ট সংগ্রহ করতে হয়।

২. সেপারেটর এবং হেডার

- **`sep`** প্যারামিটার**:** এটি ডেটা সেপারেট করার জন্য ব্যবহৃত হয়। ডিফল্ট সেপারেটর কমা। যদি ফাইলটি TSV হয় (ট্যাব দিয়ে আলাদা করা), তবে আপনাকে অবশ্যই **`sep='\t'`** ব্যবহার করে ডিফল্ট মান পরিবর্তন করতে হবে।
- কলাম নাম না থাকলে **(`names`):** যদি ডেটা সেটে কলামের নাম না থাকে এবং প্রথম রো কলামের নাম হয়ে যায়, তবে **`names`** প্যারামিটার ব্যবহার করে আপনি কলামের নামগুলির একটি কাস্টম লিস্ট দিয়ে দিতে পারেন।
- হেডার ঠিক করা **(`header`):** যদি কলামের নামগুলি সাধারণ ডেটার রো হিসেবে বিবেচিত হয়, তবে **`header=0`** সেট করলে ফাংশনটি প্রথম রো-কে কলামের নাম হিসেবে গণ্য করবে।
- ইনডেক্স কলাম **(`index_col`):** ডেটাফ্রেমের ডিফল্ট ইনডেক্সের বদলে কোনো নির্দিষ্ট কলামকে ইনডেক্স হিসেবে ব্যবহার করার জন্য এটি ব্যবহার করা হয়।

৩. ডেটা ফিল্টারিং এবং লিমিটিং

- **`usecols`:** মেশিন লার্নিংয়ের জন্য যখন সমস্ত কলামের প্রয়োজন হয় না, তখন এই প্যারামিটার ব্যবহার করে একটি লিস্টের মাধ্যমে শুধুমাত্র প্রয়োজনীয় কলামগুলির নাম উল্লেখ করা যায়। এতে অপ্রয়োজনীয় ডেটা আমদানি করার সময় বাদ হয়ে যায়।
- **`squeeze`:** যদি আপনি কেবল একটি কলাম ইম্পোর্ট করেন, তবে এটি `True` সেট করলে আউটপুটটি ডেটাফ্রেমের বদলে একটি **Pandas Series** অবজেক্ট হবে।
- **`skiprows`:** এটি দিয়ে আপনি নির্দিষ্ট রো-এর ইনডেক্স লিস্ট আকারে পাস করে দিতে পারেন, যাতে সেই রো-গুলি ডেটা লোড হওয়ার সময় বাদ পড়ে যায়। এছাড়াও, ফাংশন ব্যবহার করে জটিল লজিক (যেমন নির্দিষ্ট রো-এর মাল্টিপল বাদ দেওয়া) প্রয়োগ করা যায়।
- **`nrows`:** এটি ইম্পোর্ট করা রো-এর সংখ্যা সীমিত করে। যখন আপনার কাছে লক্ষ লক্ষ রো সহ বিশাল ডেটা সেট থাকে যা একসাথে RAM-এ লোড করা যায় না, তখন এটি বিশেষভাবে দরকারী।

৪. এনকোডিং ও এরর হ্যান্ডলিং

- **`encoding`:** ডিফল্ট এনকোডিং হলো **UTF-8**। যদি ডেটা লোড করার পর অদ্ভুত অক্ষর (gibberish characters) দেখা যায়, তবে বুঝতে হবে এনকোডিং ভিন্ন (যেমন: `latin-1`), এবং সেটি `encoding` প্যারামিটারে উল্লেখ করতে হবে।
- ত্রুটিপূর্ণ লাইন এড়িয়ে যাওয়া **(Bad Lines):** কিছু রো-তে যখন প্রত্যাশিত কলামের চেয়ে কম বা বেশি ভ্যালু থাকে, তখন পার্সিং এরর আসে। এই ক্ষেত্রে, ত্রুটিপূর্ণ লাইনগুলি এড়িয়ে যাওয়ার জন্য **`error_bad_lines=False`** বা সমতুল্য প্যারামিটার ব্যবহার করা যেতে পারে।
- ডেটা টাইপ পরিবর্তন **(`dtype`):** আপনি এই প্যারামিটার ব্যবহার করে কলামের ডিফল্ট ডেটা টাইপ পরিবর্তন করতে পারেন (যেমন: ফ্লোটকে ইন্টিজারে রূপান্তর করা), যা মেমরি বাঁচাতে সাহায্য করে।
- ডেট পার্স করা **(`parse_dates`):** ডেট কলামগুলিকে ডিফল্টভাবে স্ট্রিং হিসেবে লোড করা হয়। **`parse_dates`** প্যারামিটার ব্যবহার করে সেই কলামগুলিকে `datetime64` ফরম্যাটে রূপান্তর করা হয়। এই ফরম্যাটেই ডেট সম্পর্কিত অপারেশন (যেমন মাস বা বছর দিয়ে ফিল্টারিং) করা সম্ভব।
    - একাধিক কলাম (মাস, দিন, বছর) একত্রিত করে একটি সিঙ্গেল ডেট কলাম তৈরি করাও সম্ভব।
- কাস্টম ফাংশন **(`converters`):** এই প্যারামিটার ব্যবহার করে ডেটা লোড করার সময়ই নির্দিষ্ট কলামের উপর একটি কাস্টম পাইথন ফাংশন প্রয়োগ করা যায়। যেমন, পুরো নামকে সংক্ষেপে (যেমন: "Royal Challengers Bangalore" কে "RCB"-তে) রূপান্তর করা।

৫. মিসিং ভ্যালু এবং বিশাল ডেটা সেট

- মিসিং ভ্যালু **(`na_values`):** এটি ব্যবহার করে আপনি ডেটা সেটের মধ্যে থাকা নির্দিষ্ট স্ট্রিংগুলিকে (যেমন: "N/A" বা "---") মিসিং ভ্যালু **(NaN)** হিসেবে চিহ্নিত করতে পারেন। এটি pandas-এর ডেটা পরিষ্কার করার ফাংশনগুলিকে (যেমন `fillna` বা `dropna`) সঠিকভাবে কাজ করতে সাহায্য করে।
- বিশাল ডেটা সেট পরিচালনা **(`chunksize`):** যদি ডেটা সেট এত বড় হয় যে তা একবারে মেমরিতে লোড করা যায় না, তবে **`chunksize`** প্যারামিটার ব্যবহার করে ডেটা সেটকে নির্দিষ্ট আকারের ছোট ছোট অংশে (chunks) ভাগ করা হয়। এই প্যারামিটারটি একটি ইটারেটর রিটার্ন করে, এবং আপনাকে লুপ ব্যবহার করে প্রতিটি অংশকে আলাদাভাবে প্রসেস করতে হয়।

`read_csv` ফাংশনটি একটি বহুমুখী টুলের মতো, যা আপনাকে ডেটা লোড করার সময় ডেটা পরিষ্কার এবং কাস্টমাইজ করতে সাহায্য করে, বিশেষত যখন আপনাকে জটিল বা বিশাল ফাইল নিয়ে কাজ করতে হয়।