সূচিপত্র

ডাটা সায়েন্সের ভিত্তি

কোর্স পরিচালনায়
নুহিল মেহেদী

স্বয়ংক্রিয় কন্ট্রিবিউটরের তালিকা
(প্রথম ৫ জন)

[010]
[001]
[001]

প্রারম্ভিকা
খুব সহজ ভাষায় যদি বলা হয় তবে - ডাটা সায়েন্স হচ্ছে এরকম একটা বিশেষ জ্ঞান যার মাধ্যমে বিভিন্ন রকমের, গোছালো বা অগোছালো বিশাল পরিমাণ ডাটা থেকে সঠিক এবং অন্তর্নিহিত ব্যবহার উপযোগী তথ্য বের করে আনা যায় (এটাকে অনেকেই ডাটা মাইনিং-ও বলে থাকেন)। পরিসংখ্যান, ডাটা অ্যানালাইসিস ও সে সম্পর্কিত বিভিন্ন মেথডের সমন্বয়ে এমন একটি কনসেপ্ট যার মাধ্যমে কোন ডাটা কালেকশনের মধ্যেকার আসল ঘটনা বা বিষয় বের করে আনা যায়। এই বিজ্ঞান বস্তুত অন্যান্য অনেক ফিল্ড থেকে বিভিন্ন তত্ত্ব এবং টেকনিককে ফলো করে কাজ করে। যেমন - গণিত, পরিসংখ্যান, ইনফরমেশন সায়েন্স, কম্পিউটার সায়েন্স মেশিন লার্নিং, ক্লাস্টার অ্যানালাইসিস, ডাটা মাইনিং, ডাটাবেইজ, ডাটা ভিজুয়ালাইজেশন ইত্যাদি। কঠিন করে বলতে গেলে আরও কঠিন হয়ে যাবে। যেহেতু আমরা এই কোর্সে খুব সহজ ভাষায় ডাটা সায়েন্সের মূল ভিত্তি বিষয়ক কিছু ব্যাসিক টপিকের উপর আলোচনা করবো, তাই গুরুগম্ভীর সংজ্ঞায় না যাওয়াই ভালো। বরং, এই কোর্স থেকে একটা আবছা ধারনা নিয়ে পাঠক নিজে থেকেই পরবর্তীতে বিভিন্ন সোর্স অবলম্বন করে আরও গভীর ভাবে এই বিষয়ে পড়াশুনা করতে পারবেন।

অনেকেই ডাটা সায়েন্টিস্ট এবং পরিসংখ্যানবিদের মধ্যে পার্থক্য করতে চান না। তাই তাদের উদ্দেশ্য একটা মজার সংজ্ঞা এখানে দেয়া যেতে পারে - "Data Scientist: Person who is better at statistics than any software engineer and better at software engineering than any statistician!" :)

দিন দিন ব্যবসা, বিজ্ঞান, গবেষণা, সমাজ ব্যবস্থা, চিকিৎসা, রাজনীতি, মহাকাশবিজ্ঞান ও অনেক রকম ফিল্ডে ডাটা সায়েন্সের প্রয়োজন বেড়েই চলেছে। প্রয়োজন বাড়লেও অনেক বিশাল পরিমাণ ডাটা নিয়ে কাজ করে যথাযথ ফলাফল বা সিদ্ধান্ত আনার জন্য যে পরিমাণ অভিজ্ঞ লোক প্রয়োজন সেটা বর্তমানে নেই। ডাটা (বিশেষ করে বিগ ডাটা) নিয়ে যারা কাজ করেন, তাদেরকে বেশ কয়েকটি ভাগে ভাগ করা যায় যেমন - ডাটা ইঞ্জিনিয়ার, ডাটা সায়েনটিস্ট, স্ট্যাটিসটিসিয়ান, ডাটা অ্যানালিস্ট। অনেকেই ইদানীং মনে করছেন দিন দিন যেভাবে ডাটা বাড়ছে সে অনুযায়ী সেই ডাটা গুলো থেকে যথাযথ প্রায়োগিক ফলাফল বের করে আনার মত উপযুক্ত ডাটা প্রফেশনালের অভাবটাই এখন বড় চ্যালেঞ্জ। ডাটার প্রাপ্তি বা কম্পিউটেশন পাওয়ার চ্যালেঞ্জ এর বিষয় নয়।

আসলেই বিগ ডাটা তৈরি হচ্ছে কিভাবে? খেয়াল করলে দেখবেন - দিন দিন মানুষ সবকিছু ডিজিটালাইজ করে ফেলছে। ফেসবুক স্ট্যাটাস থেকে শুরু করে ফটো, লেখা, খবর। সিনেমা থেকে শুরু করে গবেষণার ফল, জরিপ, বিভিন্ন সেন্সর থেকে প্রাপ্ত তথ্য ইত্যাদি ইত্যাদি। বলে শেষ করা যাবে না। এমনকি, পূর্বের জমা হওয়া অ্যানালগ ডাটা গুলোকেও ডিজিটাল রূপ দেয়া হচ্ছে জোড়ে সোরে। IBM এর গবেষণা মতে, বর্তমান পৃথিবীর শতকরা ৯০ ভাগ ডিজিটাল ডাটা তৈরি হয়েছে মাত্র গত ২/৩ বছরে। তার মানে, এই ডাটা বাড়ার পরিমাণ দিন দিন জ্যামিতিক হারে বাড়তেই থাকবে। এই লিঙ্কের ইনফগ্রোফটি দেখতে পারেনঃ http://bit.ly/2r4JwYS একই সাথে এই বিশাল পরিমাণ ডাটার যথাযথ ব্যবহার নিশ্চিত করতে প্রযুক্তিগত উন্নয়নও হচ্ছে উল্লেখ যোগ্য হারে। যেমন - মেশিন লার্নিং, ডিপ লার্নিং এর মাধ্যমে এরকম বিগ ডাটা গুলোকে সঠিকভাবে ব্যবহার করে ডাটার মধ্যেকার প্যাটার্ন খোজা, ক্লাসিফাই করা, ভ্যালু প্রেডিক্ট করা ইত্যাদি কাজ এখন খুবি স্বাভাবিক। এর মাধ্যমে উক্ত ডাটা সম্পর্কিত ফিল্ড গুলো দ্রুত সিদ্ধান্ত গ্রহণ, ভবিষ্যৎ প্রেডিকশন ও অ্যানালাইসিস এর কাজ করতে পারছে সহজে যেগুলো পক্ষান্তরে উক্ত ফিল্ড গুলোকে উন্নয়নের দিকে নিয়ে যাচ্ছে।

সহজ উদাহরণ দিয়ে বুঝতে চাইলে - ধরুন একটা সুপার শপে প্রতিদিন হাজার হাজার ট্র্যাঞ্জেকশন হয়। আবার সেই কেনা বেচার মধ্যে হাজার হাজার আইটেম বিদ্যমান। আবার মনে করুন, সেই সুপার শপের বিভিন্ন লোকেশনে বিভিন্ন ব্র্যাঞ্চ আছে। সব মিলে প্রতিদিন কয়েক লাখ ট্র্যাঞ্জেকশন ঘটে এই ব্র্যান্ডের মোট বেচাকেনায়। এভাবে কয়েকমাস গেলেই যে পরিমাণ ডাটা এই স্টোরের ডাটাবেইজে তৈরি হয় তা কি নিতান্তই মুনাফা হিসাব করা আর স্টক ম্যানেজ করার মধ্যেই সীমাবদ্ধ থাকবে? যদি তাই হয় তাহলে এতো ডাটার মিস-ইউজ ছাড়া আর কিছুই করা হচ্ছে না। বরং, এই ডাটা গুলোকে যদি সঠিকভাবে পর্যালোচনা করে সেখান থেকে বিভিন্ন মজার তথ্য বের করে আনা সম্ভব হয় তাহলে ওই ব্যবসাকে আরও আধুনিক এবং যুগোপযোগী করা সম্ভব।

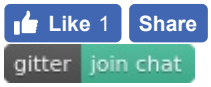
একটি উদাহরণ দেয়া যাক - একজন ক্রেতা কোন কোন আইটেম মোটামুটি একই সাথে কিনছেন শুধু এটুকু যদি ট্র্যাক করা যায় তাহলে বড় আকারের সুপার শপে ওই আইটেম গুলো পাসাপাশি সাজিয়ে রাখা যেতে পারে। এতে করে ক্রেতা খুশি হবে এবং বিক্রিও বাড়বে। আবার মনে করুন - অনলাইন স্টোরের ক্ষেত্রে একজন ক্রেতা একবার একটা জিনিষ কিনলে তাকে আরেকটা জিনিষ কেনার জন্য সাজেশন দেয়া। এটা করতে কি কি করা যেতে পারে? ধরুন ওই ক্রেতা একটা মাত্র জিনিষ কিনলো। সাথে সাথে আগের অন্যান্য ক্রেতাদের ডাটা অ্যানালাইসিস করে বের করা সিদ্ধান্তকে আমরা কাজে লাগাতে পারি। আগের অ্যানালাইসিস মোতাবেক আমাদের সিস্টেম জানে যে, বেশিরভাগ ক্রেতাই যখন এই আইটেমটা কিনেছিল তখন তারা আরেকটা আইটেমও কিনেছিল। তো, সেই আইটেমকে সাজেশন হিসেবে দেখানো যেতে পারে এই নতুন ক্রেতার কাছে। এমনকি, যদি কোন ক্রেতা কিছুই না কিনে প্রথমবার একটি সাইট ভিজিট করে সেক্ষেত্রেও আগের অ্যানালাইটিক্যাল বা প্রেডিকশন মডেল বিক্রেতাকে সাহায্য করতে পারে। যেমন - ভিজিটর কোন এলাকা থেকে ভিজিট করছে, তার বয়স কত ইত্যাদি জানা সহজ এবং যদি সিস্টেমের কাছে এরকম কিছু ক্লাসিফিকেশন ডাটা থাকে যে, ওই লোকেশনের, এই বয়সের মানুষ সব চেয়ে কোন জিনিষগুলো বেশি কিনছে তাহলেই হয়ে গেলো। এ তো, গেল খুব সহজ এবং হালকা কিছু উদাহরণ। সঠিকভাবে ডাটা সায়েন্সের প্রয়োগ কল্পনার অতীত ফলাফল এনে দিতে পারে।

ওপেন সোর্স

এই বইটি মূলত স্বেচ্ছাশ্রমে লেখা এবং বইটি সম্পূর্ন ওপেন সোর্স । এখানে তাই আপনিও অবদান রাখতে পারেন লেখক হিসেবে । আপনার কন্ট্রিবিউশান গৃহীত হলে অবদানকারীদের তালিকায় আপনার নাম স্বয়ংক্রিয়ভাবে যুক্ত হয়ে যাবে।

এটি মূলত একটি গিটহাব রিপোজিটোরি যেখানে এই বইয়ের আর্টিকেল গুলো মার্কডাউন ফরম্যাটে লেখা হচ্ছে । রিপোজিটরিটি ফর্ক করে পুল রিকুয়েস্ট পাঠানোর মাধ্যমে আপনারাও অবদান রাখতে পারেন । বিস্তারিত দেখতে পারেন এই ভিডিওতে Video

> বর্তমানে বইটির কন্টেন্ট বিভিন্ন কন্ট্রিবিউটর এবং নানা রকম সোর্স থেকে সংগৃহীত এবং সংকলিত।



ডাটা কি?

প্রতি মুহূর্তে দুনিয়াতে যা ঘটছে সবই ডাটা বা তথ্য। সহজ না? ফেসবুক স্ট্যাটাস দিচ্ছেন, ইউটিউবে ভিডিও আপলোড করছেন, কোথাও রেজিস্ট্রেশন করছেন, কোন জরিপ করছেন, দৈনিক ঘটে যাওয়া কিছু ঘটনা রেকর্ড করে রাখছেন এসবই ডাটা। আবার গবেষকগণ তাদের গবেষণার বিভিন্ন ধাপে নানা রকম তথ্য পাচ্ছেন এবং সেগুলোর লগ রাখছেন, আবহাওয়া অধিদপ্তর প্রতিদিনকার তথ্য কোথাও জমা করছে, দুর্গম কোন এলাকায় ডেপলয় করা কোন সেন্সর বা রোবট ডাটা সেন্স করে রেকর্ড করে যাচ্ছে এসবও ডাটা। কোন সুপার শপে ঘটে যাওয়া সব ট্র্যাঞ্জেকশন, ব্যাংকে ঘটমান বিভিন্ন ক্রেতা বিক্রেতার ট্র্যাঞ্জেকশন, অনলাইনে ক্রেডিট কার্ড ইউজ করে কেনা কাটা এসবও ডাটা। আরও উদাহরণ লাগবে?

সিরিয়াস কথা হচ্ছে - datum ল্যাটিন শব্দ থেকেই Data শব্দের উৎপত্তি। datum কিন্তু সিঙ্গুলার ফর্ম। data হচ্ছে এর প্লুরাল ফর্ম। তো, datum মানে হচ্ছে সিঙ্গেল কোন এন্টিটি বা সিঙ্গেল কোন একটা ঘটনার অবস্থান(বিন্দু)। এজন্য datum কে data points বলা হয়। তার মানে, data দিয়ে আসলে অনেক গুলো data points কেই বোঝানো হয়। টেকনিক্যালি Data কে Dataset হিসেবেও লেখা হয়। তাই Dataset মানেও হচ্ছে কিছু Data Point এর কালেকশন। যাই হোক খুশির খবর হচ্ছে, বর্তমানে Data শব্দকে একবচন বা বহুবচন দুভাবেই প্রকাশ করা হয়। ঝামেলা কম।

আবার বলি, ডাটা হচ্ছে কালেকশন অফ ফ্যাক্টস যেমন নাম্বার, শব্দ, পরিমাণ, পর্যবেক্ষণ এমনকি কোন কিছুর বর্ণনা। দুরকম ডাটা আছে - কোয়ালিটেটিভ ও কোয়ান্টিটেটিভ। আমার অনেক টাকা আছে, ওর চুল অনেক লম্বা; এসব কোয়ালিটেটিভ ডাটার উদাহরণ। দ্বিতীয় প্রকারের ডাটা আবার দু রকম হয় - ডিসক্রিট এবং কন্টিনিউয়াস। আমার দুটো পা, তার কাছে ১০০ টাকা আছে এগুলো ডিসক্রিট এবং সে ৫৬৫ মিলিমিটার লম্বা, আজ ২৩ মিমি বৃষ্টি হয়েছে এসব কন্টিনিউয়াস ডাটার উদাহরণ।

ডাটার ধরন
ডাটার কিছু বৈশিষ্ট্য আছে যেগুলো নিচের মত -

১) অনেক বিশাল পরিমাণে হতে পারে - আর তাই এসব অ্যানালাইসিসের জন্য ঠিক করা অ্যাল্গরিদমকে স্কেল্যাবল হতে হবে। নাহলে দেখা যাবে আপনার অ্যালগরিদম কম ডাটার উপর ঠিকি দ্রুত কাজ করতে পারে কিন্তু বেশি ডাটা নিয়ে হিসাব করতে গেলেই হ্যাং হয়ে বসে থাকে। (কমপ্লেক্সিটি অফ অ্যালগরিদম এর দরকার মনে পরে যাবে)

২) হাই ডাইমেনশনালিটি - ডাটা হতে পারে হাজার হাজার ডাইমেনশন সম্পন্ন। হুম হাজার হাজার।

৩) খুবি জটিল প্রকৃতির - যেমন সেন্সর ডাটা, বিভিন্ন ডাটা স্ট্রিম (সাউন্ড), টাইম সিরিজ ডাটা, টেম্পরাল ডাটা, সিকোয়েন্স ডাটা ইত্যাদি। মাল্টিমিডিয়া ডাটা, টেক্সট বা ওয়েব ডাটা। গ্রাফ ডাটা বা সোশাল নেটওয়ার্ক ডাটা ইত্যাদি ইত্যাদি।

> এতো ডাটা নিয়ে আমরা কি করিবো?

ভূমিকাতেই বলা আছে কি কি করা সম্ভব। আবার ধারনাতে নাই এমন কিছুও করা সম্ভব।

ডাটা মাইনিং

এতদিন শুনেছেন খনি খুরে শুধু দামি দামি জিনিসপত্র তুলে আনা হয়। তাই মাইনিং মানেই মনে হয় যে - অনেক মূল্যবান সম্পদ আহরণের কথা বলা হচ্ছে। বাস্তবে যেমন সাধারণ দেখতে একটা মরুভূমির অতল গহিনে জমে থাকতে পারে তেল, কয়লা, সোনা সহ আরও নানা রকম মহা মূল্যবান জিনিষ পত্র। তেমনি অগোছালো ডাটার মধ্যেও লুকিয়ে থাকতে পারে খুবি মূল্যবান কোন তথ্য। তাই এই বিজ্ঞানে এটাকেও মাইনিং বলা হয়।

ডাটা সায়েন্স এর সাথে ডাটা মাইনিং ওতপ্রোতভাবে জড়িত এবং একটা আরেকটার পরিপূরক। তাই এই অবস্থায় ডাটা মাইনিং এর প্রসঙ্গ নিয়ে আসা।

যাই হোক, ডাটা মাইনিং এর পুঁথিগত সংজ্ঞা হচ্ছে এরকম - "Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data."

বুঝতে পারছি :)

এর অনেক বিকল্প নামও থাকতে পারে যেগুলো শুনে ঘাবড়ানোর কিছু নাই। যেমন - Knowledge discovery (mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence ইত্যাদি। এখন বুঝলেন তো? সব হচ্ছে নামের বাহার। ঘটনা তেমন কিছু না।

KDD বা নলেজ ডিসকভারি প্রসেস

এই প্রসেসের কিছু গুরুত্ব পূর্ণ ধাপের বর্ণনা নিচে দেয়া হল -

প্রথমেই কোন ডাটাবেইজ থেকে ডাটা উদ্ধার করা হবে ->
অতঃপর সেই ডাটা গুলোকে ক্লিন করা হবে অর্থাৎ ডাটার মধ্যেকার ভুল, মিসিং ডাটা ইত্যাদি ঠিক ঠাক করা হবে ->
এরপর সেই পরিষ্কার ডাটা কে ডাটা অয়্যারহাউজে জমা করা হবে অর্থাৎ যেখান থেকে পরবর্তী ধাপে ব্যবহার করা যাবে ->
এরপর ওই ডাটা স্টোর থেকে শুধুমাত্র আমাদের উদ্দেশ্য সাধনের প্রেক্ষিতে যে ডাটা গুলো লাগবে সেগুলোকে বেছে নেয়া হবে যাকে বলে টাস্ক রেলিভেন্ট ডাটা বেছে নেয়া ->
এরপর বস্তুত ডাটা মাইনিং ঘটে বিভিন্ন অ্যালগরিদম বা টেকনিকের মাধ্যমে ->
শেষে যে প্যাটার্ন বা মূল্যবান তথ্য পাওয়া যাবে সেটাকে এভালুয়েট বা বিচার/পর্যবেক্ষণ করা হবে

কি করছি
ডাটার ভিউ মানে হচ্ছে - কি ডাটা নিয়ে কাজ করছি তা ঠিক থাকতে হবে, কি নলেজ (সম্পদ) উদ্ধার করার জন্য কাজে নামলাম সেটা ঠিক থাকতে হবে, কি টেকনিক আপ্লাই করে এই কর্ম সম্পাদন করা হবে তাও ঠিক রাখতে হবে এবং কোন সেক্টরে এই উদ্ধারকৃত সম্পদ কাজে লাগানো হবে সেটাও পরিষ্কার থাকতে হবে।

ডাটা মাইনিং ফাংশন
১) জেনারেলাইজেশন - ডাটা ক্লিনিং, ট্রান্সফরমেশন, ইন্টিগ্রেশন বা ডাটা অয়্যারহাউজ তৈরি ইত্যাদি কাজ ২) প্যাটার্ন ডিসকভারি ৩) ক্লাসিফিকেশন ৪) ক্লাস্টার অ্যানালাইসিস ৫) আউটলায়ার এনালাইসিস ৬) টাইম ও অর্ডারিং ৭) স্ট্রাকচার এনালাইসিস

প্রয়োজনীয় টুলস ও সেটআপ

সত্যি বলতে পুরো ডাটা সায়েন্স নিয়ে কাজ করতে গেলে নানা রকম টুল, সফটওয়্যার, প্ল্যাটফর্ম ইত্যাদির সাহায্য লাগতেই পারে। এর কোন নির্দিষ্ট সীমা পরিসীমা নাই। কখন কি লাগবে সেটা অবস্থাই বলে দিবে। আমরা যেহেতু এই কোর্সে ডাটা সায়েন্সের ভিত্তিটা নিয়ে আলোচনা করবো যাকে বলে intuition, তাই আমরা বিস্তর জিনিসপত্র ইন্সটল দেয়ার কথা বলে পাঠককে নিরুৎসাহিত করবো না। যখন যেটার প্রয়োজন আসবে তখন সেই টুল নিয়ে কথা বলে সেটার ইন্সটলেশন নিয়ে কথা বলা যাবে।

প্রথমেই আমাদের কম্পিউটারের সাহায্য লাগবে। তাই না? নাহলে এতো এতো ডাটা নিয়ে কি খাতা কলমে কাজ করতে চাচ্ছেন? আর কম্পিউটারকে দিয়ে কাজ করিয়ে নিতে হলে তার ভাষায় তাকে ইন্সট্রাকশন দিতে হবে। এটা তো জানেনই। তাই এই ভাষা হিসেবে আমরা আপাতত Python প্রোগ্রামিং ল্যাঙ্গুয়েজ ব্যবহার করবো। Python প্রোগ্রামিং সম্বন্ধে পড়াশুনা করতে এই কোর্সটি এখনি পড়া শুরু করতে পারেন - বাংলায় পাইথন

তো আপনার কম্পিউটারে পাইথন ইন্সটল করে নিন (উপড়ে উল্লেখিত কোর্সেই বিস্তারিত লেখা আছে)। সাথে কাজের সুবিধার্থে Jupyter Notebook ইন্সটল করে নিন। এর মাধ্যমে একটা ডকুমেন্টকে ওয়েব ব্রাউজারে ওপেন করে সেখানে বিভিন্ন কিছু লিখে এবং আলোচনা করে করে আগানো যায় এবং একটি ডকুমেন্টে বিভিন্ন ব্লক বা সেল তৈরি করে সেগুলোর মধ্যে কোড লেখা যায়। আবার আগের ধাপ বা সেলে রান করা কোড পরের ধাপেও অ্যাক্সেস করা যায়। এমনকি বাংলা লেখা এবং কোড সাথে ফটো বা ডিসপ্লে মিলিয়ে তৈরি হওয়া একটি কম্প্যাক্ট পেইজকে অন্য কারো সাথে সহজে শেয়ারও করতে পারবেন। সেই লোক তার কম্পিউটারে প্রথমে পাইথন এবং তারপর Jupyter Notebook ইন্সটল দিয়ে Jupyter Notebook এর মাধ্যমেই সেই ডকুমেন্টকে রান করাতে পারবেন। অবশ্যই উনি চাইবেন একটু পড়ে এবং একটু করে কোড রান করে কাজের প্রগ্রেস সম্বন্ধে স্বচ্ছ ধারনা নিতে। আর এখানেই Notebook এর উপকার খেয়াল করার মত। jupyter Notebook সম্পর্কে ধারনা না থাকলে একটু অন্য কোথাও থেকে আপাতত দেখে আসতে পারেন। এটা তেমন কিছু না। একটা ওয়েব অ্যাপ। এতে করে ব্রাইজারের মধ্যে একটা পেজে কোড এবং বাংলা ইংলিশ মিলিয়ে লেখা যায় এবং কোড গুলোকে রানও করা যায়। আর পুরো ডকুমেন্টের রানটাইম একটাই থাকে। এটাও খুব সহজে প্যাকেজ আকারেই ইন্সটল করা যায় এবং একটা কমান্ড দিয়েই রান করানো যায়।

আবারও বলি, Jupyter Notebook একটি ওপেন সোর্স ওয়েব অ্যাপ্লিকেশন যার মাধ্যমে লাইভ কোড (ইনস্ট্যান্ট রানেবল), কমেন্ট, ভিজুয়ালাইজেশন, ফর্মুলা ইত্যাদি মিশিয়ে ইন্ট্যারেক্টিভ ডকুমেন্ট তৈরি, শেয়ার এবং রান করা যায়।
jupyter notebook এর ফাইল এক্সটেনশন .ipynb
ডকুমেন্ট তথা ফাইলটি অন্য কাউকে আমি শেয়ার করলে সে পুরোটা ধাপে ধাপে পড়ে বুঝতে পারবে। আর যদি তার jupyter notebook ইন্সটল করা থাকে তাহলে সেখানে এই ফাইলকে ইম্পোরট করে কোড গুলো আপডেট ও রান করে দেখতে পারে।

এখনো না বুঝে থাকলে আর নিচে পড়ার দরকার নাই। আগে ঘুরে আসতে হবে এখানে

আপাতত বিভিন্ন টুল বলতে পাইথনের জন্য সহজলভ্য বেশ কিছু লাইব্রেরী আমাদের ইন্সটল করে নিতে হবে। যেমন আপাতত - numpy এবং matplotlib ইন্সটল করে নিন। খুব সহজে আপনার পাইথন এনভায়ারনমেন্টের সাথে ইন্সটল করে নিতে কমান্ড ইস্যু করতে পারেন - `pip install numpy matplotlib` অর্থাৎ এভাবে আপনি পাইথনের অফিসিয়াল প্যাকেজ ইনডেক্স থেকে pip টুলের মাধ্যমে নির্দিষ্ট কোন প্যাকেজ ইন্সটল করে নিতে পারবেন। ইন্সটল হওয়া প্যাকেজকে পরবর্তীতে আপনার পাইথন প্রোগ্রামে ব্যবহার করতে পারবেন।

numpy কি?

পাইথনে সায়েন্টিফিক কম্পিউটিং এর জন্য অতি প্রয়োজনীয় একটি লাইব্রেরী। অনেকে একে সাধারণ ভাবে ক্যালকুলেশন লাইব্রেরী বা ম্যাট্রিক্স লাইব্রেরীও বলে থাকে। লিনিয়ার অ্যালেজেব্রা, ফুরিয়ার ট্রান্সফর্ম, রেন্ডোম নাম্বার জেনারেশন এর সুবিধা সহ N-dimesional Array অবজেক্ট সাপোর্ট আছে। এতে করে অনেক জটিল ক্যালকুলেশন কম কোড লিখেই করা যায়।

```
import numpy as np
a = np.array([2,3,4])
a
```

```
array([2, 3, 4])
```

আরও একটি উদাহরণ যেখানে 15 টি এলিমেন্ট তৈরি করে সেগুলোকে 3x5 (রোxকলাম) স্ট্রাকচারে কনভার্ট করা হয়েছে ম্যাট্রিক্স এর মত করে পরবর্তীতে ব্যবহার করার জন্য।

```
a = np.arange(15).reshape(3, 5)
a
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

উপরের অ্যারে-র চেহারা চেক করা যেতে পারে নিচের মত করে,

```
a.shape
```

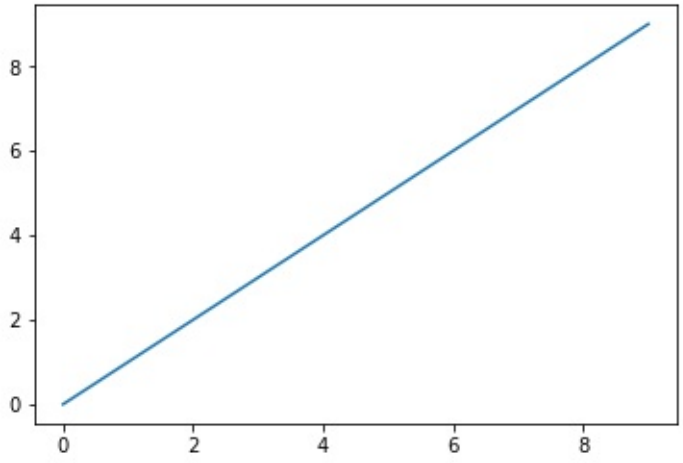
```
(3, 5)
```

পাইথনের ডিফল্ট ডাটা স্ট্রাকচার গুলো দিয়ে যে কাজ গুলো করা প্রায় অসম্ভব এবং সময় সাপেক্ষ সেগুলোর জন্য এই লাইব্রেরী বিশাল এক সমাধান।

matplotlib কি?

matplotlib এর মাধ্যমে প্রায় সব রকম গ্রাফিক্যাল প্লটিং করা যায়। নিচে এর একটি সহজ ব্যবহার দেখানো হয়েছে।

```
%matplotlib inline
from matplotlib import pyplot as plt

x = np.array(range(10))
y = np.array(range(10))

plt.plot(x, y)
plt.show()
```



mean, median, mode

mean কে অনেকেই গড় নামেই চিনে থাকবেন। খুব সহজ - যতগুলো এলিমেন্ট নিয়ে কাজ করা হচ্ছে সেগুলোর যোগফলকে মোট এলিমেন্ট সংখ্যা দিয়ে ভাগ করলেই গড় পাওয়া যায়।

```
a = np.array([10, 5, 12, 3])
np.mean(a)
```

```
7.5
```

অর্থাৎ,————

```
a = np.array([[1, 2], [3, 4]])
np.mean(a, axis=1)
```

```
array([ 1.5,  3.5])
```

অর্থাৎ, প্রথমে 1 ও 2 এর গড় এবং তারপর 3 ও 4 এর গড় করে আরেকটা অ্যারে তে জমা করা হয়েছে। numpy ব্যবহার না করলে এখানে লুপ, যোগ, ভাগ সহ বেশ কিছু কোড লিখতে হত।

median বা মধ্যক হচ্ছে কিছু ক্রমানুসারে সাজানো এলিমেন্টের মাঝখানের ভ্যালুটি অথবা মাঝখানে একাধিক ভ্যালু হলে তাদের সাধারণ গড় মানটি

```
a = np.array([10, 14, 4, 7, 9, 12, 15])
np.median(a)              # 4, 7, 9, 10, 12, 14, 15. এখানে 10 median
```

```
10.0
```

উপরের অ্যারের mean -ও বের করে দেখি,

```
a = np.array([10, 14, 4, 7, 9, 12, 15])
np.mean(a)
```

```
10.142857142857142
```

mode বা প্রচুরক হচ্ছে কোন ডাটা কালেকশনে যে এলিমেন্টটি সবচেয়ে বেশি সংখ্যক বার থাকে সেটা

```
from scipy import stats # এটি আরেকটি প্রয়োজনীয় লাইব্রেরী
a = np.array([10, 14, 4, 7, 9, 12, 4, 15]) # 4 এর উপস্থিতি বেশি
stats.mode(a)
```

```
ModeResult(mode=array([4]), count=array([2]))
```

mean থাকতে আবার median কেন?

মাঝে মাঝে কোন একটা ডাটা সেটের mean তার সঠিক/বাস্তবিক গড় প্রকাশ করে না। যেমন - নিচে কিছু লোকের বয়সের একটা অ্যারে আছে এবং এর mean এসেছে 33.84. এটা যথেষ্ট লজিক্যাল একটা ভিউ দিচ্ছে ডাটা সেট সম্পর্কে।

```
ages = np.array([30, 30, 30, 20, 20, 45, 35, 35, 30, 40, 40, 40, 45])
np.mean(ages)
```

```
33.846153846153847
```

কিন্তু ধরা যাক, সেই ডাটা সেটের মধ্যে একজন মাত্র অতিবৃদ্ধ লোকের বয়স যুক্ত করা হল যার বয়স 120 বছর। এতে করেই এই ডাটা সেটের mean বেড়ে গিয়ে হয়ে গেলো 40 যা একদমই এই সেটের বাস্তবিক গ্রহণযোগ্য ভিউকে রীতিমত বদলে ফেলেছে।

```
ages = np.array([30, 30, 30, 20, 20, 45, 35, 35, 30, 40, 40, 40, 45, 120])
np.mean(ages)
```

```
40.0
```

আবার এই অবস্থাতেও উক্ত সেটের median আসছে 35 অর্থাৎ একটা অসঙ্গতি পূর্ণ ডাটা এলিমেন্ট যুক্ত হবার পরেও median দিয়ে গড়ের একটা সঠিক ওভারভিউ পাওয়া যাচ্ছে। এরকম ক্ষেত্রে median উপকারী।

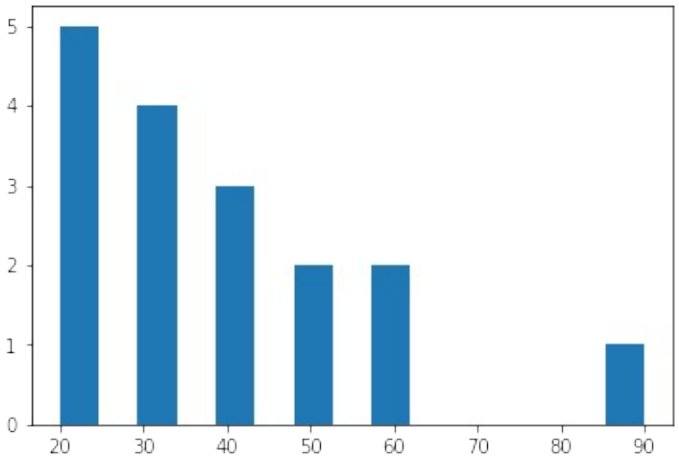
```
ages = np.array([30, 30, 30, 20, 20, 45, 35, 35, 30, 40, 40, 40, 45, 120])
np.median(ages)
```

```
35.0
```

Normal Distribution কি

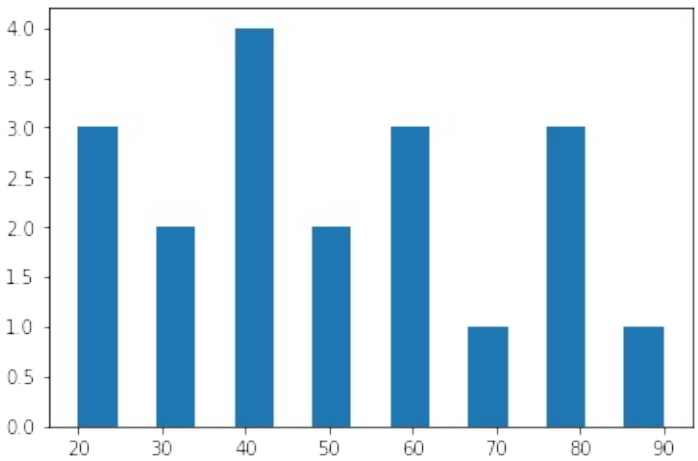
যখন আমরা কোন রিয়েল লাইফ ডাটা কালেকশনকে রিপ্রেসেন্ট/ডিস্ট্রিবিউট (স্প্রেড আউট) করি তখন সেটার চেহারা বিভিন্ন রকম হতে পারে। যেমন নিচের ডাটাসেটের হিস্টোগ্রাম শো করলে দেখা যাচ্ছে বাম দিকে লম্বা বার বেশি,

```
x = np.array([20, 30, 30, 30, 20, 20, 50, 20, 20, 30, 40, 40, 40,  50, 60, 60, 90])

plt.hist(x, 15)
plt.show()
```



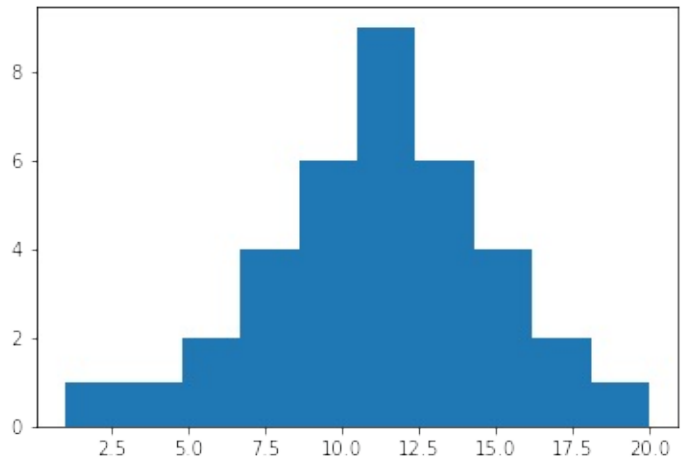
অথবা কিছু ডাটার হিস্টোগ্রাম বার গুলো হতে পারে নিচের মত অগোছালো,

```
x = np.array([40, 30, 80, 30, 20, 80, 50, 20, 20, 60, 40, 40, 40,  50, 60, 60, 90, 80,
 70])

plt.hist(x, 15)
plt.show()
```

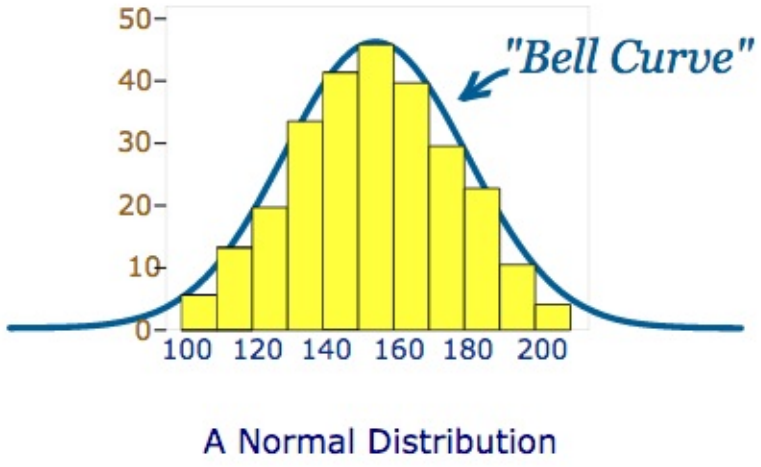


কিন্তু অনেক সময় বাস্তবের কিছু ডাটাকে (কিছু ছাত্রের উচ্চতার মান, তাদের পরীক্ষায় প্রাপ্ত নম্বর, একটি মেশিন দারা তৈরি কোন প্রোডাক্টের সাইজ, জনগণের আয় ইত্যাদি) ডিস্ট্রিবিউট করলে নিচের মত চেহারা পাওয়া যায়,

```
sizes = np.array([9, 8, 8, 9, 9, 1, 4, 5, 6, 7, 8, 9, 10, 10, 11, 11, 11, 11, 11, 13,
12, 12, 12, 12, 13, 13, 14, 14, 14, 15, 15, 15, 16, 17, 18, 20])

plt.hist(sizes)
plt.show()
```



যেটা অনেকটা বেল (ঘণ্টি) কার্ভের মত অর্থাৎ মাঝখানের বার গুলো লম্বা এবং তার দুপাশের বার গুলো ক্রমান্বয়ে ছোট। এরকম কোন ডাটার ডিস্ট্রিবিউশনকে বলা হয় নরমালি ডিস্ট্রিবিউটেড।



সব ডাটা এমনি এমনি এমন চেহারা নাও পেতে পারে। সেক্ষেত্রে ডাটা গুলোর গড় বা মধ্যক বের করে সেটাকে মাঝখানে রেখে ওই মধ্যম মানের চেয়ে ছোট ও বড় মান গুলোকে যথাক্রমে বাম পাশে এবং ডানপাশে রেখে একটি ডিস্ট্রিবিউশন তৈরি করাকে নরমাল ডিস্ট্রিবিউশন বলা হয়। ডাটাকে এভাবে ডিস্ট্রিবিউট করলে পরবর্তীতে অনেক রকম হিসাব, পর্যবেক্ষণ বা সম্ভাব্যতা বের করা সহজ হয়ে যায়।

নরমালি ডিস্ট্রিবিউটেড কোন ডাটাসেটের mean, median এবং mode মোটামুটি একই হয়। নিচে প্রমাণ করে দেখা যেতে পারে,

```
np.mean(sizes)
```

```
11.194444444444445
```

```
np.median(sizes)
```

```
11.0
```

```
stats.mode(sizes)
```

```
ModeResult(mode=array([11]), count=array([5]))
```

ভ্যারিয়েন্স ও স্ট্যান্ডার্ড ডেভিয়েশন

আমরা আগেই বলেছি ডাটা ডিস্ট্রিবিউশন করাকে স্প্রেড আউট করা বা ছড়িয়ে দেয়াও বলা যায়। সেক্ষেত্রে আমরা জানতে পেরেছি যে নরমালি ডিস্ট্রিবিউটেড ডাটা বা ডাটাকে নরমালি ডিস্ট্রিবিউট করার অনেক সুবিধা আছে। তো, সেই নরমালি ডিস্ট্রিবিউট করার পর যদি পর্যবেক্ষণ করি যে- ডাটা গুলো গড় মান থেকে কতটা ছড়ানো বা এর থেকে কত দুরে অবস্থিত সেক্ষেত্রে যে ফ্যাক্টরটি সম্বন্ধে জানতে হবে সেটি হচ্ছে উক্ত ডিস্ট্রিবিউশনের ভ্যারিয়েন্স।

ভ্যারিয়েন্স হচ্ছে - উক্ত ডিস্ট্রিবিউশনের mean (গড়) মান থেকে প্রত্যেকটি এলিমেন্টের দূরত্বের বর্গের গড়। অর্থাৎ, উপরের sizes অ্যারের ভ্যারিয়েন্স বের করার জন্য আমরা নিচের ফর্মুলা ব্যবহার করতে পারি,

যেখানে হচ্ছে এলিমেন্ট এবং হচ্ছে গড়। আর হচ্ছে মোট এলিমেন্ট সংখ্যা।

আর, স্ট্যান্ডার্ড ডেভিয়েশন হচ্ছে ভ্যারিয়েন্স এর বর্গমূল,

নিজে নিজে ক্যালকুলেশনটা করে দেখতে পারেন। আমি numpy এর std ফাংশন ব্যবহার করে তাড়াতাড়ি জেনে নেই স্ট্যান্ডার্ড ডেভিয়েশন কত,

```
np.std(sizes)
```

```
3.9144990061482714
```

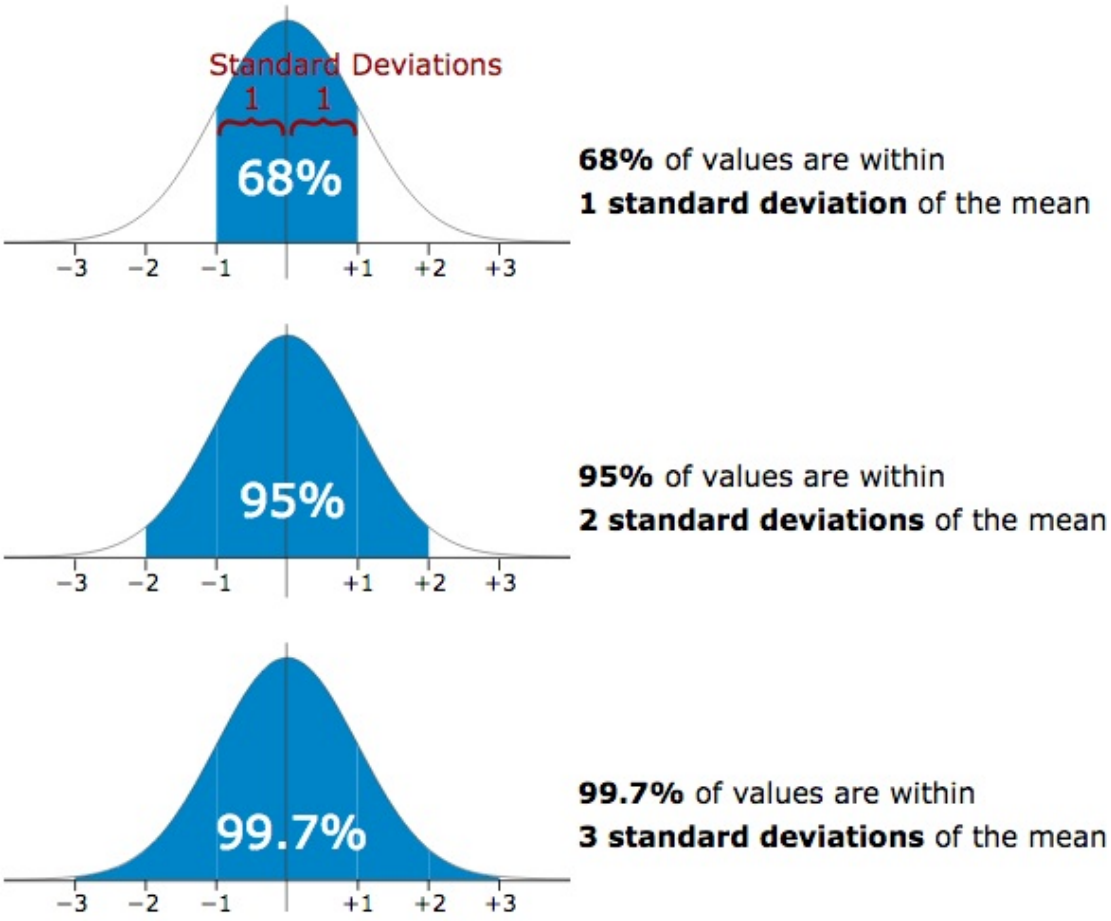
স্ট্যান্ডার্ড ডেভিয়েশন জানার উপকার

নরমালি ডিস্ট্রিবিউটেড ডাটার ক্ষেত্রে এর স্ট্যান্ডার্ড ডেভিয়েশন জানা হয়ে গেলে ওই ডাটা গুলো সম্পর্কে কিছু অনুসিদ্ধান্ত পাওয়া যায়। যেমন

- ৬৮% ডাটা গড় ভ্যালু তথা মধ্য লাইন থেকে দুই পাশে ১ একক পরিমাণ স্ট্যান্ডার্ড ডেভিয়েশনের মধ্যে থাকে
- ৯৫% ডাটা গড় ভ্যালু তথা মধ্য লাইন থেকে দুই পাশে ২ একক পরিমাণ স্ট্যান্ডার্ড ডেভিয়েশনের মধ্যে থাকে
- ৯৯% ডাটা গড় ভ্যালু তথা মধ্য লাইন থেকে দুই পাশে ৩ একক পরিমাণ স্ট্যান্ডার্ড ডেভিয়েশনের মধ্যে থাকে

অর্থাৎ, আমরা যদি উপরের sizes অ্যারে থেকে শতকরা ৯৫% ডাটা নিয়ে কাজ করতে চাই অথবা শতকরা ৯৫% ডাটাকে গ্রহণযোগ্য ডাটা মনে করে থাকি তাহলে কোন কোন ডাটা আমরা নেব এবং কোন গুলো ফেলে দেব সেটা খুব সহজেই জানতে পারি। অনুসিদ্ধান্ত থেকে আমরা জানি গড় মান থেকে ডান এবং বাম পাশে ২ একক পরিমাণ স্ট্যান্ডার্ড ডেভিয়েশনের মধ্যেই এই পরিমাণ ডাটা থাকার কথা।

একে **Probability Distribution** ও বলা হয়ে থাকে



এখন গড় মান থেকে আমরা প্রত্যেকটি এলিমেন্ট এর দূরত্ব হিসাব করে দেখবো স্ট্যান্ডার্ড ডেভিয়েশন এককে। যেগুলো সামনে ও পেছনে ২ একক পরিমাণ স্ট্যান্ডার্ড ডেভিয়েশনের মধ্যে থাকবে শুধু সেগুলোকেই নেব। উদাহরণ সরূপ,

গড় 11.19
স্ট্যান্ডার্ড ডেভিয়েশন 3.91
অ্যারের প্রথম এলিমেন্ট 9
যেহেতু 9 গড় মানের বাম দিকে অবস্থান করে তাই গড় থেকে স্ট্যান্ডার্ড ডেভিয়েশন বিয়োগ করে এর অবস্থান চেক করে দেখবো,
গড় - ২ একক স্ট্যান্ডার্ড ডেভিয়েশন = 11.19 - (2 * 3.91) = 4.08
অর্থাৎ 9 এর অবস্থান ২ একক স্ট্যান্ডার্ড ডেভিয়েশন মধ্যেই আছে। এই এলিমেন্টকে আমরা গ্রহণ করবো

গড় 11.19
স্ট্যান্ডার্ড ডেভিয়েশন 3.91
অ্যারের আরেকটি এলিমেন্ট 16
যেহেতু 16 গড় মানের ডান দিকে অবস্থান করে তাই গড় থেকে স্ট্যান্ডার্ড ডেভিয়েশন যোগ করে এর অবস্থান চেক করে দেখবো,
গড় + ২ একক স্ট্যান্ডার্ড ডেভিয়েশন = 11.19 + (2 * 3.91) = 19.72
অর্থাৎ 16 এর অবস্থান ডান দিকে ২ একক স্ট্যান্ডার্ড ডেভিয়েশন মধ্যেই আছে। এই এলিমেন্টকেও আমরা গ্রহণ করবো

গড় 11.19
স্ট্যান্ডার্ড ডেভিয়েশন 3.91
অ্যারের আরেকটি এলিমেন্ট 1
যেহেতু 16 গড় মানের বাম দিকে অবস্থান করে তাই গড় থেকে স্ট্যান্ডার্ড ডেভিয়েশন বিয়োগ করে এর অবস্থান চেক করে দেখবো,
গড় - ২ একক স্ট্যান্ডার্ড ডেভিয়েশন = 11.19 - (2 * 3.91) = 4.08
অর্থাৎ 1 এর অবস্থান বাম দিকে ২ একক স্ট্যান্ডার্ড ডেভিয়েশনেরও বাইরে (বামে)। তাই এই এলিমেন্টকে আমরা গ্রহণ করবো না কারন এটা আমাদের পছন্দের শতকরা ৯৫ ভাগ স্বাভাবিক ডাটা-র মধ্যেও পরে না।

একই ভাবে 20 ও বাদ পরে যাবে কারন এটি বেল কার্ভের অতিরিক্ত ডান দিকে অবস্থান করছে।

স্ট্যান্ডার্ডাইজেশন এবং এর প্রয়োজনীয়তা

একটি এলিমেন্ট আলোচ্য নর্মাল ডিস্ট্রিবিউশনের গড় মান থেকে কত একক স্ট্যান্ডার্ড ডেভিয়েশন দূরত্বে অবস্থান করে সেই ভ্যালুকে উক্ত এলিমেন্টের স্ট্যান্ডার্ড স্কোর, সিগমা বা z-score বলে। এভাবে একটি ডাটাকে z-score এ কনভার্ট করাকেই স্ট্যান্ডার্ডাইজেশন বলে।

সূত্র, z-score = (যে ভ্যালুর স্ট্যান্ডার্ডাইজেশন করতে হবে - গড়)/ স্ট্যান্ডার্ড ডেভিয়েশন

এটা করার ফলে বিশেষ কিছু সময়ে সুবিধা পাওয়া যায়। যেমন উদাহরণ সরূপ - একজন শিক্ষক ১১ জন ছাত্রের পরীক্ষা নিলেন ৬০ নম্বরের মধ্যে। পাশ মার্ক বলে দিলেন ৩০। কিন্তু পরীক্ষা এতোই কঠিন হল যে ১০ জন ছাত্রই ৩০ এর চেয়ে কম মার্ক পেল। এখন সবাইকে তো আর ফেইল করে দেয়া যায় না। তাই তিনি ঠিক করলেন মার্কস গুলোকে স্ট্যান্ডার্ডাইজ করবেন এবং যারা ১ একক স্ট্যান্ডার্ড ডেভিয়েশনের নিচে মার্ক পেয়েছে শুধু তাদেরকে ফেইল করে দিবেন। এতে করে তিনি শতকরা ৬৮% ভালো ছাত্র বেছে নিতে পারছেন।

mean বের করে ফেলি,

```
marks = np.array([20, 15, 26, 32, 18, 28, 35, 14, 26, 22, 17])
np.mean(marks)
```

```
23.0
```

স্ট্যান্ডার্ড ডেভিয়েশন

```
np.std(marks)
```

```
6.6332495807107996
```

প্রথম ছাত্রের

অতএব, সূত্র অনুযায়ী উপরের মার্কস গুলোর স্ট্যান্ডার্ড স্কোর যথাক্রমে,
-0.45, -1.21(fail), 0.45, 1.36, -0.76, 0.76, 1.82, -1.36(fail), 0.45, -0.15, -0.91

অর্থাৎ শুধুমাত্র 15 এবং 14 মার্ক পাওয়া ছাত্র দুজনকে ফেইল করে দিতে পারেন তিনি।

একটু বেশি ডাটা নিয়ে কাজ

এখন পর্যন্ত আমরা ম্যানুয়ালি একদম অল্প কিছু ডাটা নিয়ে শুধুমাত্র ম্যাথেমিটিক্যাল টার্ম গুলো বোঝার চেষ্টা করেছি। এখন থেকে আমরা একটু বেশি সংখ্যক ডাটার উপর কাজ করবো যাতে করে ফ্যাক্টর গুলোর সঠিকটা আরও ভালভাবে যাচাই করা যায়। এ জন্য আমরা numpy এর রেন্ডমাইজেশন ফাংশন এর সাহায্য নিয়ে চাহিদা মোতাবেক বিভিন্ন ডাটাসেট বানিয়ে সেগুলোর উপর পরীক্ষা চালাবো।

```
incomes = np.random.normal(27000, 15000, 10000)
np.mean(incomes)
```

```
27012.587884334778
```

উপরে আমরা একটা নরমাল ডিস্ট্রিবিউশন তৈরি করেছি যার ডাটাসেট হচ্ছে কিছু লোকের মাসিক ইনকাম। এর সেন্টার মান ঠিক করে দিয়েছি 27000, স্ট্যান্ডার্ড ডেভিয়েশন বলে দিয়েছি 15000 এবং মোট 10000 -টি ডাটা পয়েন্ট তৈরি করতে বলেছি। এই রেন্ডোম ডাটা সেটের mean তথা গড় মান বের করতে আমরা numpy এর mean ফাংশন কল করেছি এবং এর ভ্যালু এসেছে ঠিক 27000 এর মতই।

আর নিচে আমরা উক্ত ডাটা গুলোকে 50 টি সেগমেন্টে ভাগ করে একটা হিস্টোগ্রাম দেখার চেষ্টা করেছি।

```
plt.hist(incomes, 50)
plt.show()
```

```
np.median(incomes)
```

```
26976.888137643109
```

অর্থাৎ আবারও প্রমাণ হয় যে - নরমালি ডিস্ট্রিবিউটেড ডাটার ক্ষেত্রে mean, median এবং mode মোটামুটি একই।

Outlier হচ্ছে এমন ভ্যালু যেটা আলোচ্য সাধারণ ভ্যালু থেকে যথেষ্ট দুরে বা বাইরে অবস্থান করে। অর্থাৎ উপরের ইনকাম এমাউন্ট গুলোর মধ্যে যদি এমন কোন লোকের ইনকাম যুক্ত করা যায় যার মাসিক আয় 1000000000 তাহলে এটাকে আউটলায়ার বলা হয় এবং এটা অবাঞ্ছিতভাবে mean ভ্যালু বদলে দেয়।

```
incomes = np.append(incomes, [1000000000])
np.mean(incomes)
```

```
126999.88789554522
```

কিন্তু এক্ষেত্রেও median সাহায্য করে সঠিক গড় ভিউ পেতে,

```
np.median(incomes)
```

```
26977.609357910656
```

আর হ্যাঁ, স্ট্যান্ডার্ড ডেভিয়েশন জানা থাকলে কিন্তু আমরা সহজেই এরকম আউটলায়ার গুলোকে চিহ্নিত করে বাতিল করে দিতে পারি। কারন আমরা জানি সেন্টার ভ্যালু থেকে ২/৩ একক স্ট্যান্ডার্ড ডেভিয়েশনেরও বাইরে পরবে এরকম আউটলারায় গুলো। তাই এগুলোকে আনইউজুয়াল হিসেবে চিহ্নিত করা যায়। z-score এর কথা নিশ্চয়ই মনে আছে এতক্ষণেও।

পপুলেশন ও স্যাম্পল

এ দুটো যথাক্রমে Census এবং Sample এর প্রতিশব্দ। এখানে আবার এই প্রসঙ্গ আনার কারন হচ্ছে, যখন Sample নিয়ে কাজ করতে হবে তখন Sample Variance জানতে হবে যা কিনা Population নিয়ে কাজ করার সময়কার সাধারণ Variance থেকে ভিন্ন।

N সংখ্যক স্যাম্পল নিয়ে কাজ করার সময় Sample Variance বের করার সূত্রে ভগ্নাংশের নিচে মোট এলিমেন্ট (গোটা পপুলেশন) সংখ্যা না হয়ে N-1 হবে। আর স্বভাবতই Sample Standard Deviation হবে ওই Sample Variance এর Square Root.

দু ক্ষেত্রেই যথাক্রমে এবং

এ অবস্থায় আরেকটি উদাহরণ দেখে নেই,

```
incomes = np.random.normal(100.0, 50.0, 10000) # মেন্টার ভ্যালু 100, স্ট্যান্ডার্ড ডেভিয়েশন 20, ডাটা পয়েন্ট 10000 টি

plt.hist(incomes, 50)
plt.show()
```

```
incomes.var() # Variance
```

```
2483.8524780006833
```

```
incomes.std() # Standard Deviation
```

```
49.838263192056395
```