Quant trading from a beginner perspective

1. Skill sets
   1. Logical and Objective Mind
   2. Believe in a systematic trading approach
   3. You can write rules for your system in a trading diary with pen and paper
      1. My journey started with 30 30-minute congested pattern break-out style trading system that I wrote in a notepad (I wasn't the one who created it, I just wrote with my tweaks)
      2. The above leads me to Market Profile and there started to make a trading system around it
   4. Take baby steps
      1. convert readily available indicators to a trading system
         1. Backrest them and analyze the report from 3rd person's perspective

See also: [Use Readily available trading software](#Use_Readily_available_trading_software)

1. Use Readily available trading software
   1. Pros
      1. Easy learning curve
      2. No need to start everything from scratch
      3. Coding language is often easy for this trading software since it's made only for trading and not for general purpose (we are going to use the same approach for Python)
      4. can backtest and execute the system within the platform in some cases(broker's back end needs to be supported)
   2. Cons
      1. Mostly ant-free
      2. Each s/w has its limitations regarding its coding framework
         1. you can't bypass that unless s/w developer introduces the support you needed
         2. You can't use another s/w if you dedicated enough time to learn one s/w
      3. These s/w have an agreement with external data providers so using your custom data is always going to be discouraged by these developers
         1. I believe half the trading edge is in the data itself, using the same data as everyone does you are limited to specific market-driven information
   3. Some s/w bypassed that by introducing general-purpose programming, notably C# but it's useful only if you have a background in that language

See also: [Each s/w has their limitations regarding their coding framework](#Each_s_w_has_their_own_limitations_regarding_their_coding_framework)

* 1. Trading Software (Big disclaimer, I am not in the loop with recent versions and new tools since I use Python for everything
     1. MultiCharts
        1. pros
           1. The oldest and well-supported platform-specific programming language called EasyLanguage or PowerLanguage
           2. Most developed backtesting framework
           3. It supports various American and Forex brokers to trade directly from their s/w so system conversion leak is minimum
           4. It supports your own CSV data with no resource hog
           5. DDE support to connect Exel to MC, you can get data from Excel and also export too, unlike other s/w DDE support (DDE is dead now and Microsoft replaced it with OLE Automation object model, don't know how many of these s/w uses that to fetch the custom data)
           6. They have a c# version as well that they call MultiCharts DotNet
        2. Cons
           1. Not Free
           2. Indian brokers are not supported to execute trades and the same for other countries apart from the US and UK

NSE (Indian Stock Market Exchange) has several restrictions for automation they made Quant Trading a paywall. Blame mostly goes to the root cause of all SEBI (governing body like SEC)

* + - * 1. There can be errors in backtesting results and you will never know why since you don't know the source code (happened with me occasionally in the past)
        2. Only for Windows O/S
        3. On low-end hardware won't run since it's not lightweight (one of the reasons why Indian traders stayed away from it during 2005-08). I think nowadays it should work fine though with a minimum of 8GB RAM and I3 level processor
    1. Market Delta (Investor RT, realized that there are several Market Delta tools, but in the video, I was talking about Investor RT)
       1. Pros
          1. Making a system in Investor RT is very easy. Just drag and drop functions
          2. great Footprint charts that you can backtest
          3. Good support (2009, don't know about now) Once they adjusted their FootPrint-related function which was limited for certain historical data)
       2. Cons
          1. Making the system was easy but very limited
          2. Drag and drop sounds easy for the learning curve but it's time-consuming in a longer time
          3. Iffy Backtesting results, would always give me different results LOL (it was version 9 in the year 2009)
          4. Resource heavy to import external data
          5. No active community to discuss MC-related ideas or issues
    2. Ninja Trader
       1. pros
          1. C# support
          2. good Footprint charts
          3. due to c# you can make a lot of customization
          4. Supports a lot of US brokers
       2. Cons
          1. If you don't have c# background then it's tough
          2. new version iteration is extremely slow, updates are very infrequent and fewer
    3. Amibroker
       1. pros
          1. Extremely light
          2. Huge community for Indian traders
          3. AFL(Amibroker programming lang) library is very big, a lot of strategies and indicators you can import
       2. cons
          1. Outside India it's non-existent
          2. No support for any broker
          3. AFL has a little steeper learning curve. At least for me who had no tech background, learned Python way faster than AFL
    4. Sierra Charts
       1. I only used it for like 1 week so I can't say much
       2. Heard that Footprint charts are the best in Sierra

1. Python
   1. What previously mentioned gets into systematic trading and not much in actual Quant trading but it's good to test water and develop the programming mindset
   2. Set your goals before learning Python
      1. Why you want to learn Python
         1. Trading only
            1. When you search Python, most results are not specifically targeted at traders which is understandable since it's the most popular general purpose coding language now And that's what makes people scared it feels like such a monumental task. Whenever you feel like the objective is huge always cut it into pieces and look at each part separately and objectively
            2. Start learning a Python data science framework called pandas which is a programming language on its own and should be enough for the beginning and then choose one data visualization model like plotly or matplotlib
            3. Back in the day Python could only be used for analytical purposes and streaming live data you had to rely on other platforms unless you mastered several hacks in matplotlib which wasn't worth the trouble for the basic looking chart. Using PLotly's Dash framework you can visualize streaming data with better customization.
         2. Trading only+ Machine Learning/AI (\*highly recommended)
            1. Trading-related Python means the stat side of Python which directly overlaps with AI, a lot of skill sets you learn in data science are useful in Machine Learning

Get familiar with Pandas DataFrame which should be more than enough during the initial stage and learn at least one deterministic data visualization library

Earlier python was used for analytical purposes only and still, It's used for that if you're transitioning from consumer-level Chart tools to Python then you will miss streaming charts but the good news is plotless dash can do that very easily and much more with huge customization scope that these charting platforms you used couldn't provide I will make a series of videos on that and its one of my favourite topic I simply love data visualization it tells a lot that pure stats can not tell and yet keeps us objective since it is you who created the chart so you know what is going on behind that visualization you know stats for nerds kind of thing that you see on YouTube

* + - * 1. What is the difference between Machine Learning and AI

usually, it is the same but most folks intuitively separate machine learning as statistical-driven models where you don't have to train the model like K means, decision Trees etc. while AI is based on deep learning where you have to train the model with tons of data, it mimics how humans learn that's why also called as neural engine

* + - * 1. what is useful for trading

I highly recommend you start with statistically driven Machine Learning models and as you improve your skills and get comfortable with data preprocessing which is the most difficult part then you can dive into deep learning

* + - 1. for general purpose so you can get into a larger Python eco-system as a professional software developer
         1. You will benefit from the bottom- approach learn the basics 1st and then develop your programming skills as in progress you need to have your basics very tight to scale your code larger it gets extremely complex and need a team of developers so you have to have every basic thing right you can always start trading anytime you want once you get to that level
  1. python installation
     1. Usually it should not be a separate topic for any software but the irony is for such a simple language the installation is always a pain in the wrong area especially on Windows or should I say only on Windows since Python comes pre-installed in Linux and Mac
        1. To make life easier always and always install Python with Anaconda software, especially for people who are interested in data science. It comes with pandas, Numpy and Scipy which are essential for data science and that too with c dependencies so you do not have to install them separately

1. Hardware
   1. CPU
      1. Mid-level
      2. AND Ryzen
         1. more power with the same buck
         2. Ryzen 5600x
      3. If you have intel
         1. upgrade if not already i5 >6th gen
         2. If not, it's not a dealbreaker but makes life easy
   2. SSD
      1. use larger memory models
      2. Improves speed dramatically, a goto upgrade for old PCs to improve speed
   3. RAM
      1. 8 GB minimum
         1. Use dual channel i.e. 2 4 GB RAMs instead of 1 8GB
      2. if 4 GB, add extra or if don't have a slot replace, 4GB is not gonna work, sorry
   4. GPU
      1. NVIDIA RTX series Mid-range. Do not buy AMD GPUs they do not support CUDA and it is essential for Machine Learning
      2. uses high-memory models. 4 GB should start
         1. Use
            1. You will use mostly for inference
            2. For training you have to spend a fortune, instead use Google's Colab

If you are serious buy premium. But the free tier should be more than enough to train your model

Don't recommend other cloud platforms since resources are limited on the internet

* 1. Desktop or laptop
     1. Always use a desktop and try to learn to make your own it is not that complicated there are 1000000 videos on how to build your rig you will have better control.
        1. If you are serious about Machine Learning then you need to understand the hardware as well. Learning about software is not enough.
           1. some of the biggest algo trading firms have a very basic strategy but extremely high-end hardware to leverage the speed and efficiency

1. Operating system
   1. Windows
      1. WSL2
         1. Sometimes it is inevitable that you come across a Python library that only supports Linux and breaks on Windows are mostly uses Linux and they could not test in such instances have to have a Linux operating system so WSL will bridge that gap
   2. Linux
      1. For a more experienced trader who wants to replicate ideas mentioned in the research paper and test them locally
   3. May 3rd recommendation
      1. Make your PC dual boot so you can boot into Windows or Linux as per the requirement it is extremely easy to do that and also it will be a nice experience to learn the Linux system environment without ditching your comfort zone or if you are an avid gamer and want to play a pc game then you can switch back to windows. I know Linux nowadays supports video games the customization needed for that is not worth the hassle for new users 4 just started to learn Linux
         1. unlike the Windows operating system there are various variances of Linux operating system and they are called Linux flavours it all boils down to variants of Ubuntu, Debian and Fedora
            1. Ubuntu is targeted at those who want to shift from Windows so they can feel at home, so Ubuntu variants are always recommended. My favourite is Linux Mint. Manjaro was also quite friendly and had the support of Arch Linux
            2. With these flavours there are desktop versions, XFCE and all and non-GUI ones too as server
2. IDE
   1. Pycharm Studio
      1. Interactive Framework
         1. easy for learning
      2. free
      3. comes with a data visualization window called Sciview, similar to Matlabs variable viewer
      4. Less need for customization, so you can focus more on coding
   2. Visual Studio Code
   3. Spyder
      1. Comes with Anaconda installation
         1. interactive framework
      2. comes with a data visualization window called Sciview, similar to Matlabs variable viewer
   4. Jupyter Lab
      1. Comes with Anaconda installation
         1. web-based
   5. Sublime Text
   6. Atom