RStudio 2020 Internship Application

Riccardo Esclapon

Contents

Chapter 1

Overview

Chapter 2

What makes me a good fit

Here are some of the things I believe make me a great fit for the internship:

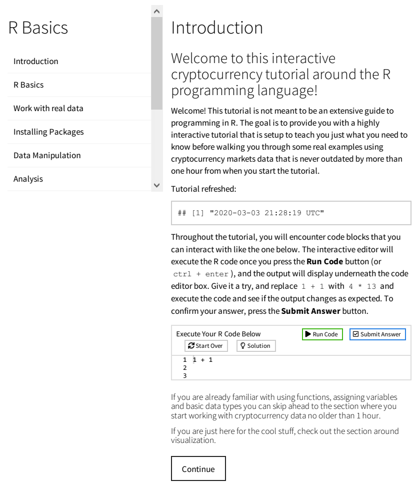
2.1 I .Rmd files

I was completely blown away by the R Markdown file format when I first discovered it, and I definitely felt like the courses I took in college in R should have mentioned the .Rmd format, as well as the tidyverse and the idea behind the pipe operator. I have spent a lot of my time learning R Markdown and digging through books and amazing resources made available by RStudio, so here are some of my favorite output formats that I am looking to teach people about:

2.1.1 Learnr

I have been using learnr for about a year and a half, and recently I started to offer programming tutorials on my website using learnr where every time the tutorial is opened, users learn to program in R using data from the cryptocurrency markets that is never outdated by more than 1 hour:

(this takes about 30 seconds to load, give it more time if it's showing up blank)



Introduction

Welcome to this interactive cryptocurrency tutorial around the R programming language!

Welcome! This tutorial is not meant to be an extensive guide to programming in R. The goal is to provide you with a highly interactive tutorial that is setup to teach you just what you need to know before walking you through some real examples using cryptocurrency markets data that is never outdated by more than one hour from when you start the tutorial.

Tutorial refreshed:

```
## [1] "2020-03-03 21:28:19 UTC"
```

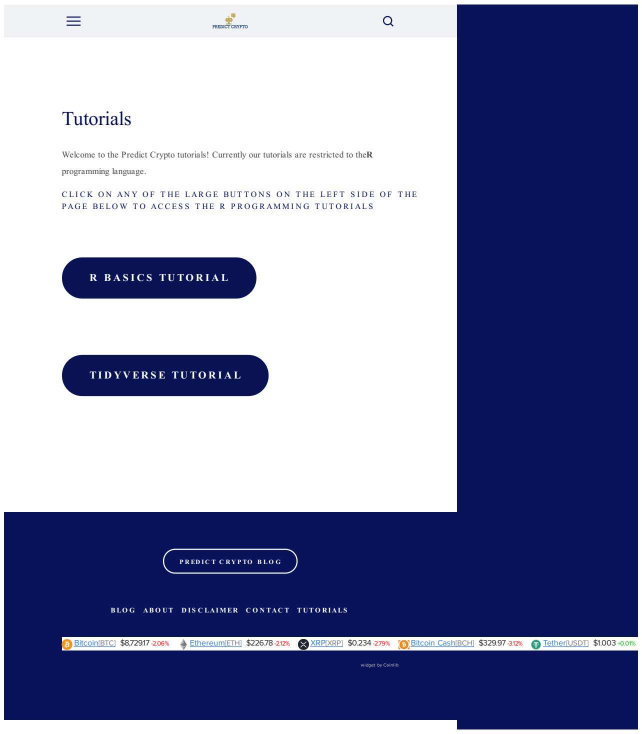
Throughout the tutorial, you will encounter code blocks that you can interact with like the one below. The interactive editor will execute the R code once you press the **Run Code** button (or `ctrl + enter`), and the output will display underneath the code editor box. Give it a try, and replace `1 + 1` with `4 * 13` and execute the code and see if the output changes as expected. To confirm your answer, press the **Submit Answer** button.

```
1 + 1
```

If you are already familiar with using functions, assigning variables and basic data types you can skip ahead to the section where you start working with cryptocurrency data no older than 1 hour.

If you are just here for the cool stuff, check out the section around visualization.

I would recommend looking at the **Visualization** section to visually see that the data is never outdated by more than 1 hour.



Tutorials

Welcome to the Predict Crypto tutorials! Currently our tutorials are restricted to the**R** programming language.

CLICK ON ANY OF THE LARGE BUTTONS ON THE LEFT SIDE OF THE PAGE BELOW TO ACCESS THE R PROGRAMMING TUTORIALS

Bitcoin[BTC] $8,729.17 -2.06% Ethereum[ETH] $226.78 -2.12% XRP[XRP] $0.234 -2.79% Bitcoin Cash[BCH] $329.97 -3.12% Tether[USDT] $1.003 +0.01%

widget by Coinlib

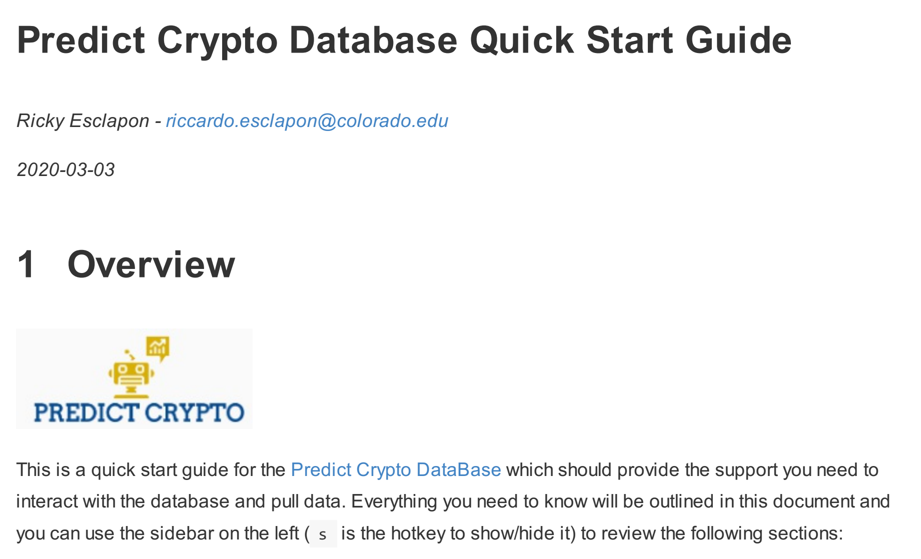
I post these on my website:

I'm loving the integrated tutorials tab within RStudio in the 1.3 preview and I am working towards including these with my `PredictCrypto` package, which I talk more about and use in the next section of this document.

2.1.2 Bookdown

I was very close to paying for a monthly subscription on gitbook.com because I thought it was such an amazing format to provide documentation through, so I was particularly impressed by and grateful for the bookdown (Xie, 2020) package, and these days it's my go to for organizing most things I work on, so why not my application?

This document is obviously an example of a bookdown document in itself, but here's another guide I put together using bookdown:



Predict Crypto Database Quick Start Guide

Ricky Esclapon - riccardo.esclapon@colorado.edu

2020-03-03

1 Overview

PREDICT CRYPTO

This is a quick start guide for the Predict Crypto DataBase which should provide the support you need to interact with the database and pull data. Everything you need to know will be outlined in this document and you can use the sidebar on the left (`s` is the hotkey to show/hide it) to review the following sections:

This guide refreshes daily in order to show a preview of the latest data within the document and you can look at the GitHub Actions daily runs here. You can also see the refreshed data in the *useful tables* section of the document.

I also found that documentation done in bookdown can work really great when working within a large company as well, and I put together some very thorough documentation for a project using bookdown that was very well received (but I can't show here). In my particular case it worked really well because I could send the link to the html index of the bookdown document and when opened it would behave like a website hosted on the shared folders within the secure network which ended up being particularly simple and effective.

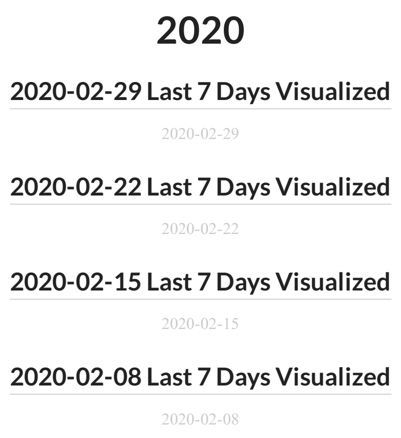
2.1.3 Presentations

I am a **big** fan of ioslides and revealjs in particular as R Markdown outputs. I find the revealjs output to be incredibly cool with the rotating cube animation, and the ability to not only move forward but move downward adds a surprisingly useful tool to break down topics; ioslides is just really clean, well made and easy to use and looks great with widescreen enabled. I aspire to be an expert in Xaringan one day but am not currently.

Making presentations in R Markdown is what really got me working with .Rmd files, because I started working towards a very specific project using an idea I haven't really seen elsewhere of creating presentations that give the user options and as they make their way through the slides, those options affect not only what they see in the slides that come afterwards, but also the options they are given. For example, the user could choose to do an analysis for a particular asset, then choose the main category of the analysis to perform, then the sub-category of the analysis and so on, until by the end of the presentation the user has performed an analysis that was completely unique and tailored to their preferences and interests. See the gif below for an example of what this looks like:

2.1.4 Blogdown

Blogdown(Xie, 2019) and bookdown work very similarly, so most of what I mentioned in the bookdown section applies here. Because my website predictcrypto.com only shows the latest data based on the current date, I leverage blogdown to create weekly snapshots of the visualizations over the last 7 day period: https://predictcryptoblog.com/.


2020

2020-02-29 Last 7 Days Visualized

2020-02-29

2020-02-22 Last 7 Days Visualized

2020-02-22

2020-02-15 Last 7 Days Visualized

2020-02-15

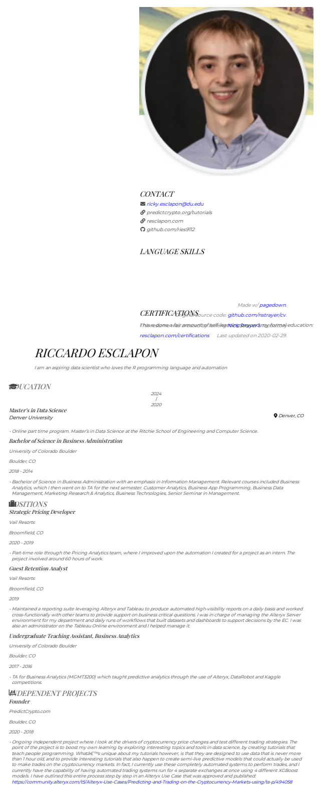
2020-02-08 Last 7 Days Visualized

2020-02-08

Because all these systems work so well with automation, as I keep adding new interesting content to my website I can also add archives of that content using blogdown.

2.1.5 Pagedown

Pagedown(Xie et al., 2020) is yet another awesome way to create html outputs and I used Nick Strayer's repository https://github.com/nstrayer/cv to build my cv and resume using his template:



CONTACT

ricky.esclapon@du.edu

predictcrypto.org/tutorials

resclapon.com

github.com/ries9112

LANGUAGE SKILLS

Made w/ pagedown.

CERTIFICATIONS

Original source code: github.com/nstrayer/cv.

I have done a fair amount of self-learning beyond my formal education:

This resume was created by forking Nick Strayer's repository

resclapon.com/certifications

Last updated on 2020-02-29.

RICCARDO ESCLAPON

I am an aspiring data scientist who loves the R programming language and automation

EDUCATION

2024
/
2020

Master's in Data Science
Denver University

Denver, CO

- Online part time program. Master's in Data Science at the Ritchie School of Engineering and Computer Science.

Bachelor of Science in Business Administration

University of Colorado Boulder

Boulder, CO

2018 - 2014

- Bachelor of Science in Business Administration with an emphasis in Information Management. Relevant courses included Business Analytics, which I then went on to TA for the next semester. Customer Analytics, Business App Programming, Business Data Management, Marketing Research & Analytics, Business Technologies, Senior Seminar in Management.

POSITIONS

Strategic Pricing Developer

Vail Resorts

Broomfield, CO

2020 - 2019

- Part-time role through the Pricing Analytics team, where I improved upon the automation I created for a project as an intern. The project involved around 60 hours of work.

Guest Retention Analyst

Vail Resorts

Broomfield, CO

2019

- Maintained a reporting suite leveraging Alteryx and Tableau to produce automated high-visibility reports on a daily basis and worked cross-functionally with other teams to provide support on business critical questions. I was in charge of managing the Alteryx Server environment for my department and daily runs of workflows that built datasets and dashboards to support decisions by the EC. I was also an administrator on the Tableau Online environment and I helped manage it.

Undergraduate Teaching Assistant, Business Analytics

University of Colorado Boulder

Boulder, CO

2017 - 2016

- TA for Business Analytics (MGMT3200) which taught predictive analytics through the use of Alteryx, DataRobot and Kaggle competitions.

INDEPENDENT PROJECTS

Founder

PredictCrypto.com

Boulder, CO

2020 - 2018

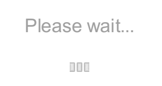
- Ongoing independent project where I look at the drivers of cryptocurrency price changes and test different trading strategies. The point of the project is to boost my own learning by exploring interesting topics and tools in data science, by creating tutorials that teach people programming. Whatâ€™s unique about my tutorials however, is that they are designed to use data that is never more than 1 hour old, and to provide interesting tutorials that also happen to create semi-live predictive models that could actually be used to make trades on the cryptocurrency markets. In fact, I currently use these completely automated systems to perform trades, and I currently have the capability of having automated trading systems run for 4 separate exchanges at once using 4 different XGBoost models. I have outlined this entire process step by step in an Alteryx Use Case that was approved and published: https://community.alteryx.com/t5/Alteryx-Use-Cases/Predicting-and-Trading-on-the-Cryptocurrency-Markets-using/ta-p/494058

Big thanks to Nick Strayer for the awesome template!

2.1.6 Flexdashboard

Flexdashboards (Iannone et al., 2018) were my first introduction to shiny apps and I was completely blown away by that framework and have used it for several projects and is one of my absolute favorite tools.

To get some practice, I converted some of the content found in Tidy Text Mining by Julia Silge and David Robinson and made it into a flexdashboard. **I made no changes to the code found within the book**, this was simply an experiment to learn more about flexdashboards and semantic analysis:


Please wait...

2.2 Automation

Automation is at the center of everything I do and my one true passion. One of my big goals for RStudio::conf 2020 was to learn more about automating things through GitHub using CI since I always had a hard time figuring that out, and the things I learned about especially relating to GitHub actions and using Netlify were above my expectations in terms of the ease of use, capabilities and free tier offerings, and I am super excited to share how crazy simple automating a very complex process can be through RStudio, GitHub Actions and Netlify.

I didn't fnd a huge wealth of information on automating things in R through GitHub Actions and I'm excited to share those learnings in the months to come.

It's pretty mindblowing that these frameworks allow a user to create an interactive book with complex javascript, HTML, CSS, TeX, etc... from scratch, deploy it to an https secured website and create an automated process around it, all in less than 10 minutes with minimal code involved. What's even more powerful, is that the same methodologies can be applied to make other interfaces and outputs, like making a blogdown website, and I can't speak highly enough of all the work Yihui blessed us all with.

I have also done a lot of automation work for Vail Resorts using a tool called Alteryx to create fully automated processes with the main purpose of refreshing Tableau dashboards offering refreshed datasets relating to ski pass sales. You can find an example of an automated Alteryx process I created for a personal project doing automated trading on the cryptocurrency markets using my own database, SQL, R and Python here: https://community.alteryx.com/t5/Alteryx-Use-Cases/Predicting-and-Trading-on-the-Cryptocurrency-Markets-using/ta-p/494058

2.3 Fit Within the Company

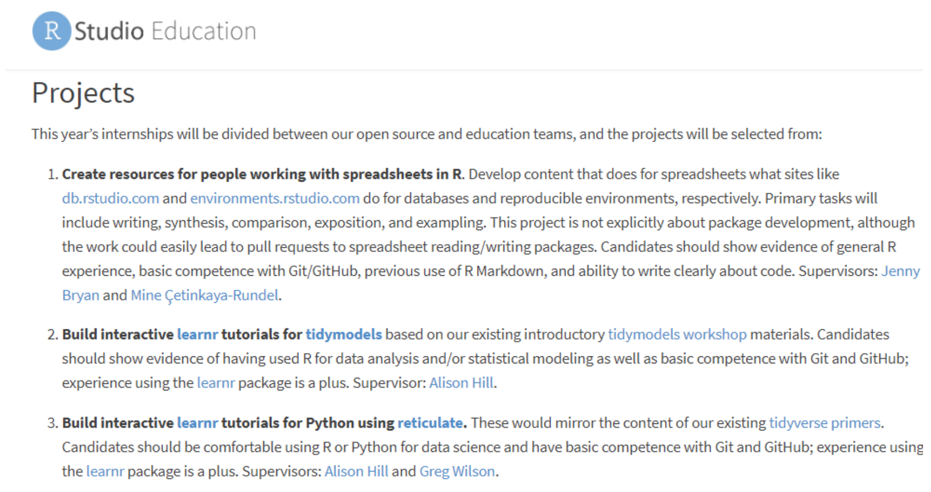
After following along with the RStudio::conf 2019 as it was happening, I knew I had to make it out to RStudio::conf 2020, and it was a truly incredible experience. I learned everything I was hoping to learn about and then some, and JJ's talk and BCorp announcement really resonated with me. Generally speaking my philosophy is that the most straightforward way to success is to help other people succeed, and I believe I share the values that RStudio holds dear as a company. The content of JJ's talk around the model that companies currently operate under, the pursuit of profit being a legal obligation, how we got to this point, and the need for this model to evolve, was inspirational. I am **very** impressed by RStudio taking a strong stance in this area and I agree with JJ's message wholeheartedly.

Another thing I was really impressed with was the focus and clarity around why everything should be reproducible and how not making your work publicly available can often be very costly in many different ways. RStudio has done an incredible job at making powerful and complex frameworks easy for anyone to use. I am driven to making easy to follow and informative content to help other *"fellow self-taught programmers who were told they weren't good enough but are too driven and excited to care"* (Gans, 2020) in the same spirit of former superstar interns like Maya. I also want to give Maya a shoutout for her amazing tidyblocks project; in college I learned to use an incredible tool called Alteryx and I have become somewhat of an expert in it, but it's a publicly traded company and since I have learned to use it the price has gone up from an already ridiculous ~$3.5k a year to now being $5,195 a year, and as JJ pointed out in his talk this type of model is not a long-term sustainable model for programming software, and forget about reproducibility and outside access. The tidyblocks project works in a fundamentally different way because of the scratch-like design, but it's actually got the main pieces to replicate what Alteryx does (which I believe is actually mostly built using R when processing data), and that's been one of my main goals since tuning in for RStudio::conf 2019, so I have just started making my way through her Javascript for Data Science book and in mid-late 2021 I hope to be able to start making some contributions to that project.

I work well both in-person and remotely. I have a dedicated home office to do my independent work in, with a powerful desktop PC and two monitors. I have experience working remotely and keeping myself accountable without someone looking over my shoulder.

Chapter 3

Projects Well Suited For



RStudio Education

Projects

This year's internships will be divided between our open source and education teams, and the projects will be selected from:

1. **Create resources for people working with spreadsheets in R.** Develop content that does for spreadsheets what sites like db.rstudio.com and environments.rstudio.com do for databases and reproducible environments, respectively. Primary tasks will include writing, synthesis, comparison, exposition, and exampling. This project is not explicitly about package development, although the work could easily lead to pull requests to spreadsheet reading/writing packages. Candidates should show evidence of general R experience, basic competence with Git/GitHub, previous use of R Markdown, and ability to write clearly about code. Supervisors: Jenny Bryan and Mine Çetinkaya-Rundel.
2. **Build interactive learnr tutorials for tidymodels** based on our existing introductory tidymodels workshop materials. Candidates should show evidence of having used R for data analysis and/or statistical modeling as well as basic competence with Git and GitHub; experience using the learnr package is a plus. Supervisor: Alison Hill.
3. **Build interactive learnr tutorials for Python using reticulate.** These would mirror the content of our existing tidyverse primers. Candidates should be comfortable using R or Python for data science and have basic competence with Git and GitHub; experience using the learnr package is a plus. Supervisors: Alison Hill and Greg Wilson.

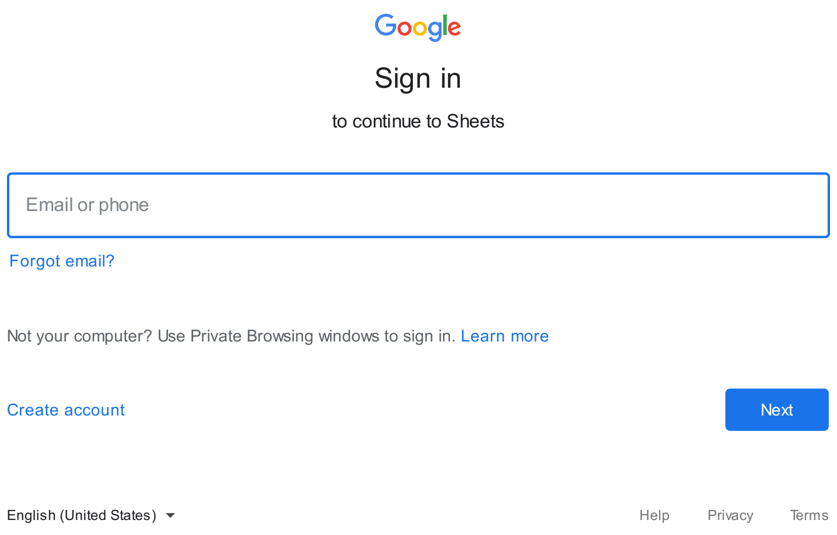
3.1 Create resources for people working with spreadsheets in R

What better way to show I am suited for a project than to give a hands-on example? See the code below for a use-case using `googlesheets4`(Bryan, 2020).

First I will go ahead and import every package in the `tidyverse`(Wickham, 2019):

```
library(tidyverse)
```

We will be importing the following spreadsheet:



```r
spreadsheet_url <- "https://docs.google.com/spreadsheets/d/1_zRBFrB1au7qhxuDDfDuh_bPLG
```

Before importing the data, let's use `tictoc` (Izrailev, 2014) to measure how long each step takes. I am using `tic()` to start the time for both the total execution time and for the step reading the data in. After importing the data we will run `toc()` to get the execution time for that step.

```r
library(tictoc)
tic('Total section 3 runtime')
tic('Read googlesheets data')
```

Now let's import the `googlesheets4` and read a spreadsheet I made for this internship application, specifying the sheet called *coinmetrics_preview* inside the function `read_sheet()`:

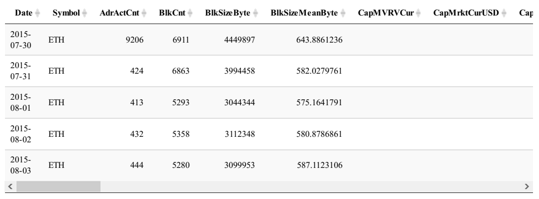
```r
library(googlesheets4)
googlesheets_data <- read_sheet(spreadsheet_url, sheet = 'coinmetrics_preview') %>% as.data.frame
```

```r
toc()
```

```
## Read googlesheets data: 44.75 sec elapsed
```

Let's take a peek at the first 1,000 rows using `DT::datatable()` (Xie et al., 2019)

```r
library(DT)
datatable(head(googlesheets_data,1000),  style = "default",
          options = list(scrollX = TRUE, pageLength=5,dom='t'), rownames = F)
```



Date	Symbol	AdrActCnt	BlkCnt	BlkSizeByte	BlkSizeMeanByte	CapMVRVCur	CapMrktCurUSD	Cap
2015-07-30	ETH	9206	6911	4449897	643.8861236			
2015-07-31	ETH	424	6863	3994458	582.0279761			
2015-08-01	ETH	413	5293	3044344	575.1641791			
2015-08-02	ETH	432	5358	3112348	580.8786861			
2015-08-03	ETH	444	5280	3099953	587.1123106			

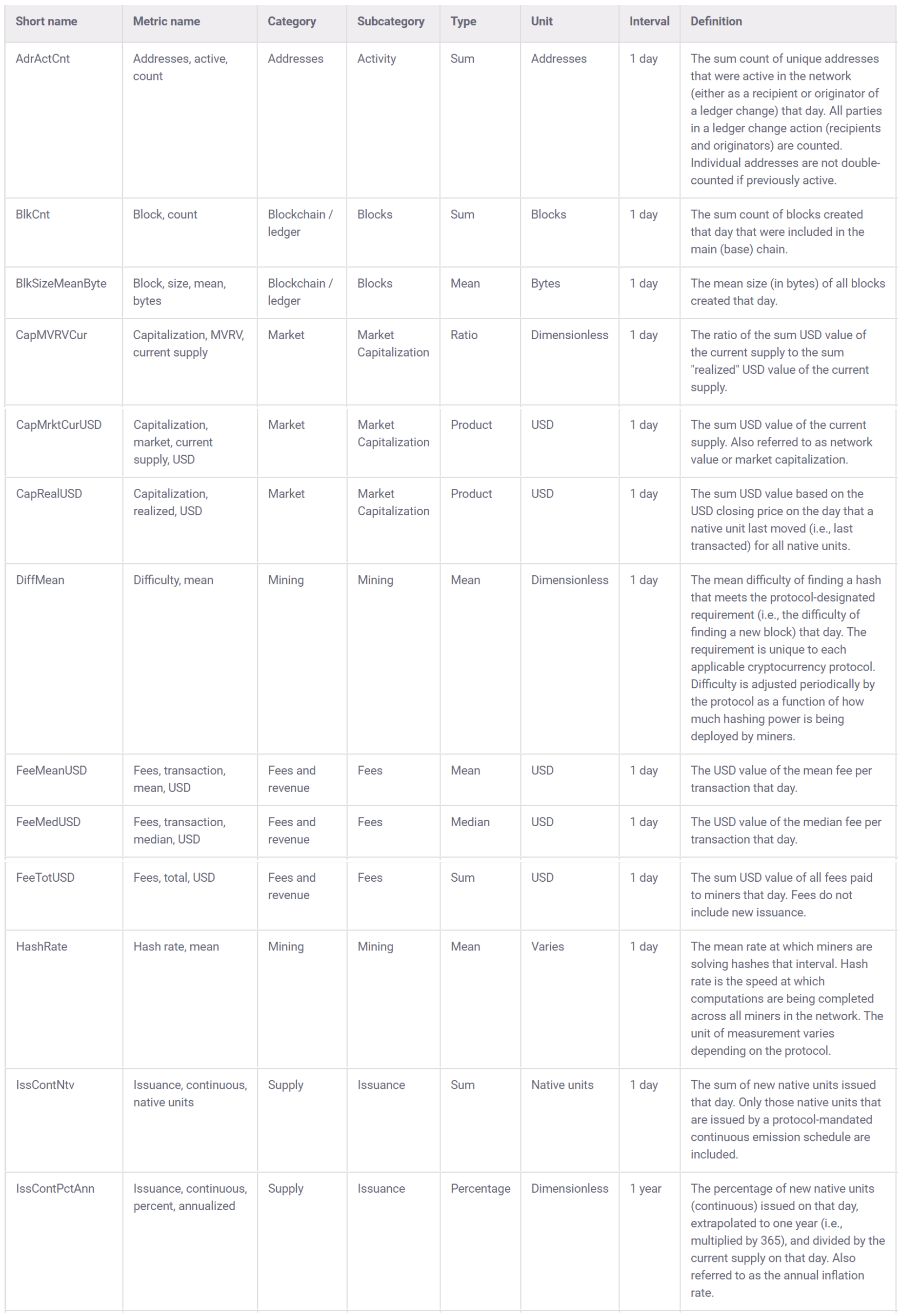
This data is sourced from the website coinmetrics.io

How many rows in the dataset?

```r
nrow(googlesheets_data)
```

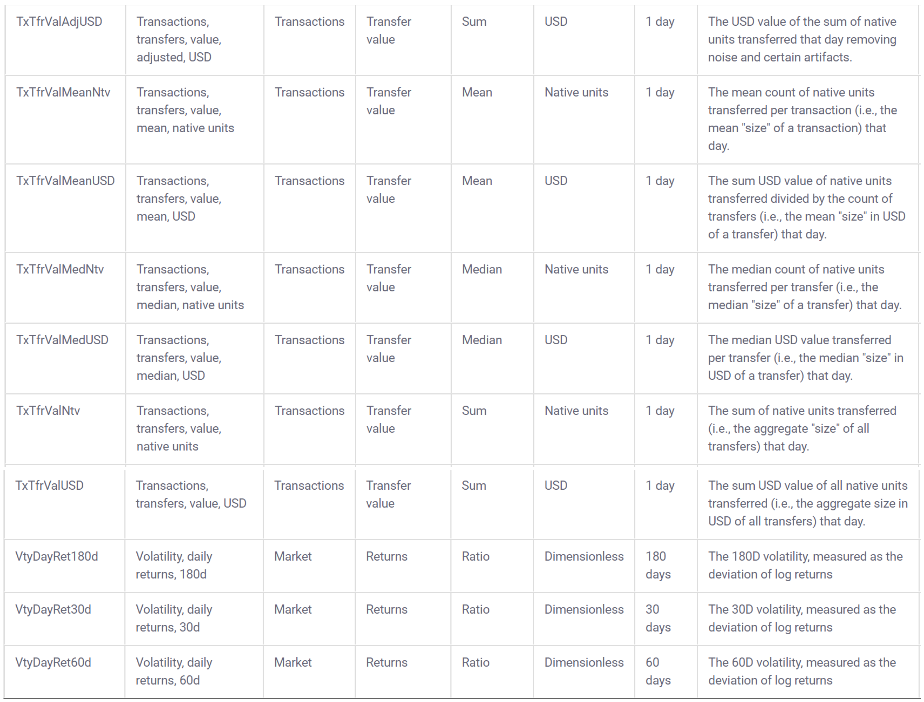
```
## [1] 11995
```

Coinmetrics also provides a data dictionary to go along with the data:



Short name	Metric name	Category	Subcategory	Type	Unit	Interval	Definition
AdrActCnt	Addresses, active, count	Addresses	Activity	Sum	Addresses	1 day	The sum count of unique addresses that were active in the network (either as a recipient or originator of a ledger change) that day. All parties in a ledger change action (recipients and originators) are counted. Individual addresses are not double-counted if previously active.
BlkCnt	Block, count	Blockchain / ledger	Blocks	Sum	Blocks	1 day	The sum count of blocks created that day that were included in the main (base) chain.
BlkSizeMeanByte	Block, size, mean, bytes	Blockchain / ledger	Blocks	Mean	Bytes	1 day	The mean size (in bytes) of all blocks created that day.
CapMVRVCur	Capitalization, MVRV, current supply	Market	Market Capitalization	Ratio	Dimensionless	1 day	The ratio of the sum USD value of the current supply to the sum "realized" USD value of the current supply.
CapMrktCurUSD	Capitalization, market, current supply, USD	Market	Market Capitalization	Product	USD	1 day	The sum USD value of the current supply. Also referred to as network value or market capitalization.
CapRealUSD	Capitalization, realized, USD	Market	Market Capitalization	Product	USD	1 day	The sum USD value based on the USD closing price on the day that a native unit last moved (i.e., last transacted) for all native units.
DiffMean	Difficulty, mean	Mining	Mining	Mean	Dimensionless	1 day	The mean difficulty of finding a hash that meets the protocol-designated requirement (i.e., the difficulty of finding a new block) that day. The requirement is unique to each applicable cryptocurrency protocol. Difficulty is adjusted periodically by the protocol as a function of how much hashing power is being deployed by miners.
FeeMeanUSD	Fees, transaction, mean, USD	Fees and revenue	Fees	Mean	USD	1 day	The USD value of the mean fee per transaction that day.
FeeMedUSD	Fees, transaction, median, USD	Fees and revenue	Fees	Median	USD	1 day	The USD value of the median fee per transaction that day.
FeeTotUSD	Fees, total, USD	Fees and revenue	Fees	Sum	USD	1 day	The sum USD value of all fees paid to miners that day. Fees do not include new issuance.
HashRate	Hash rate, mean	Mining	Mining	Mean	Varies	1 day	The mean rate at which miners are solving hashes that interval. Hash rate is the speed at which computations are being completed across all miners in the network. The unit of measurement varies depending on the protocol.
IssContNtv	Issuance, continuous, native units	Supply	Issuance	Sum	Native units	1 day	The sum of new native units issued that day. Only those native units that are issued by a protocol-mandated continuous emission schedule are included.
IssContPctAnn	Issuance, continuous, percent, annualized	Supply	Issuance	Percentage	Dimensionless	1 year	The percentage of new native units (continuous) issued on that day, extrapolated to one year (i.e., multiplied by 365), and divided by the current supply on that day. Also referred to as the annual inflation rate.

IssContUSD	Issuance, continuous, USD	Supply	Issuance	Sum	USD	1 day	The sum USD value of new native units issued that day. Only those native units that are issued by a protocol-mandated continuous emission schedule are included (i.e., units manually released from escrow or otherwise disbursed are not included).
IssTotUSD	Issuance, total, USD	Supply	Issuance	Sum	USD	1 day	The sum USD value of all new native units issued that day.
NVTAdj	NVT, adjusted	Valuation	Valuation	Ratio	Dimensionless	1 day	The ratio of the network value (or market capitalization, current supply) divided by the adjusted transfer value. Also referred to as NVT.
NVTAdj90	NVT, adjusted, 90d MA	Valuation	Valuation	Ratio	Dimensionless	1 day	The ratio of the network value (or market capitalization, current supply) to the 90-day moving average of the adjusted transfer value. Also referred to as NVT.
PriceBTC	Price, BTC	Market	Price	NA	BTC	1 day	The fixed closing price of the asset as of 00:00 UTC the following day (i.e., midnight UTC of the current day) denominated in BTC.
PriceUSD	Price, USD	Market	Price	NA	USD	1 day	The fixed closing price of the asset as of 00:00 UTC the following day (i.e., midnight UTC of the current day) denominated in USD. This price is generated by Coin Metrics' fixing/reference rate service.
SplyCur	Supply, current	Supply	Current supply	Sum	Native units	All time	The sum of all native units ever created and visible on the ledger (i.e., issued) as of that day. For account-based protocols, only accounts with positive balances are counted.
TxCnt	Transactions, count	Transactions	Transactions	Sum	Transactions	1 day	The sum count of transactions that day. Transactions represent a bundle of intended actions to alter the ledger initiated by a user (human or machine). Transactions are counted whether they execute or not and whether they result in the transfer of native units or not (a transaction can result in no, one, or many transfers). Changes to the ledger mandated by the protocol (and not by a user) or post-launch new issuance issued by a founder or controlling entity are not included here.
TxTfr	Transactions, transfers, count	Transactions	Transfers	Sum	Transactions	1 day	The sum count of transfers that day. Transfers represent movements of native units from one ledger entity to another distinct ledger entity. Only transfers that are the result of a transaction and that have a positive (non-zero) value are counted.
TxTfrValAdjNtv	Transactions, transfers, value, adjusted, native units	Transactions	Transfer value	Sum	Native units	1 day	The sum of native units transferred that day removing noise and certain artifacts.



TxTfrValAdjUSD	Transactions, transfers, value, adjusted, USD	Transactions	Transfer value	Sum	USD	1 day	The USD value of the sum of native units transferred that day removing noise and certain artifacts.
TxTfrValMeanNtv	Transactions, transfers, value, mean, native units	Transactions	Transfer value	Mean	Native units	1 day	The mean count of native units transferred per transaction (i.e., the mean "size" of a transaction) that day.
TxTfrValMeanUSD	Transactions, transfers, value, mean, USD	Transactions	Transfer value	Mean	USD	1 day	The sum USD value of native units transferred divided by the count of transfers (i.e., the mean "size" in USD of a transfer) that day.
TxTfrValMedNtv	Transactions, transfers, value, median, native units	Transactions	Transfer value	Median	Native units	1 day	The median count of native units transferred per transfer (i.e., the median "size" of a transfer) that day.
TxTfrValMedUSD	Transactions, transfers, value, median, USD	Transactions	Transfer value	Median	USD	1 day	The median USD value transferred per transfer (i.e., the median "size" in USD of a transfer) that day.
TxTfrValNtv	Transactions, transfers, value, native units	Transactions	Transfer value	Sum	Native units	1 day	The sum of native units transferred (i.e., the aggregate "size" of all transfers) that day.
TxTfrValUSD	Transactions, transfers, value, USD	Transactions	Transfer value	Sum	USD	1 day	The sum USD value of all native units transferred (i.e., the aggregate size in USD of all transfers) that day.
VtyDayRet180d	Volatility, daily returns, 180d	Market	Returns	Ratio	Dimensionless	180 days	The 180D volatility, measured as the deviation of log returns
VtyDayRet30d	Volatility, daily returns, 30d	Market	Returns	Ratio	Dimensionless	30 days	The 30D volatility, measured as the deviation of log returns
VtyDayRet60d	Volatility, daily returns, 60d	Market	Returns	Ratio	Dimensionless	60 days	The 60D volatility, measured as the deviation of log returns

3.2 Build interactive learnr tutorials for tidymodels

3.2.1 Data Prep

Using the data from coinmetrics, I will create a predictive model to forecast the percentage change in price over time.

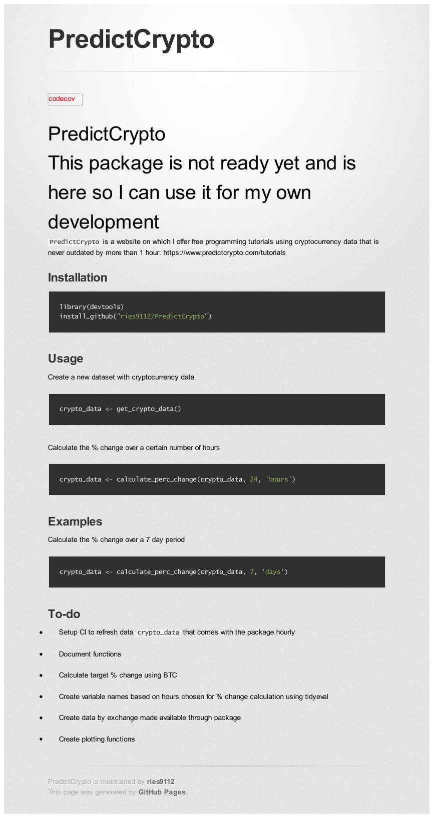
First, I will import a package that I am making that is **still in development** `PredictCrypto`:

```
library(PredictCrypto)
```

(this is an in-development tool that I will use for a research paper I am working on)

I attended the two day building tidy tools workshop working with Charlotte and Hadley at RStudio::conf 2020 and I am comfortable writing packages in R as well as using testthat and showing code coverage for a repository.

Here is the GitHub Pages environment associated with the repository:



PredictCrypto

codecov

PredictCrypto
This package is not ready yet and is here so I can use it for my own development

`PredictCrypto` is a website on which I offer free programming tutorials using cryptocurrency data that is never outdated by more than 1 hour: https://www.predictcrypto.com/tutorials

Installation

```
library(devtools)
install_github("ries9112/PredictCrypto")
```

Usage

Create a new dataset with cryptocurrency data

```
crypto_data <- get_crypto_data()
```

Calculate the % change over a certain number of hours

```
crypto_data <- calculate_perc_change(crypto_data, 24, 'hours')
```

Examples

Calculate the % change over a 7 day period

```
crypto_data <- calculate_perc_change(crypto_data, 7, 'days')
```

To-do

- Setup CI to refresh data `crypto_data` that comes with the package hourly
- Document functions
- Calculate target % change using BTC
- Create variable names based on hours chosen for % change calculation using tidyeval
- Create data by exchange made available through package
- Create plotting functions

PredictCrypto is maintained by **ries9112**
This page was generated by **GitHub Pages**.

I am going to convert the column names from ***CamelCase*** to ***snake_case*** using the `janitor`(Firke, 2020) package because the functions in my package use snake_case and I want to avoid mixing the two:

Before:

```
##  [1] "Date"             "Symbol"           "AdrActCnt"        "BlkCnt"
##  [5] "BlkSizeByte"      "BlkSizeMeanByte" "CapMVRVCur"       "CapMrktCurUSD"
##  [9] "CapRealUSD"       "DiffMean"         "FeeMeanNtv"       "FeeMeanUSD"
## [13] "FeeMedNtv"        "FeeMedUSD"        "FeeTotNtv"        "FeeTotUSD"
## [17] "HashRate"         "IssContNtv"       "IssContPctAnn"    "IssContUSD"
## [21] "IssTotNtv"        "IssTotUSD"        "NVTAdj"           "NVTAdj90"
## [25] "PriceBTC"         "PriceUSD"         "ROI1yr"           "ROI30d"
## [29] "SplyCur"          "TxCnt"            "TxTfrCnt"         "TxTfrValAdjNtv"
## [33] "TxTfrValAdjUSD"   "TxTfrValMeanNtv" "TxTfrValMeanUSD" "TxTfrValMedNtv"
## [37] "TxTfrValMedUSD"   "TxTfrValNtv"      "TxTfrValUSD"      "VtyDayRet180d"
## [41] "VtyDayRet30d"     "VtyDayRet60d"     "DateTimeUTC"
```

```r
library(janitor)
googlesheets_data <- clean_names(googlesheets_data)
```

After:

```
##  [1] "date"                 "symbol"               "adr_act_cnt"
##  [4] "blk_cnt"              "blk_size_byte"        "blk_size_mean_byte"
##  [7] "cap_mvrv_cur"         "cap_mrkt_cur_usd"     "cap_real_usd"
## [10] "diff_mean"            "fee_mean_ntv"         "fee_mean_usd"
## [13] "fee_med_ntv"          "fee_med_usd"          "fee_tot_ntv"
## [16] "fee_tot_usd"          "hash_rate"            "iss_cont_ntv"
## [19] "iss_cont_pct_ann"     "iss_cont_usd"         "iss_tot_ntv"
## [22] "iss_tot_usd"          "nvt_adj"              "nvt_adj90"
## [25] "price_btc"            "price_usd"            "roi1yr"
## [28] "roi30d"               "sply_cur"             "tx_cnt"
## [31] "tx_tfr_cnt"           "tx_tfr_val_adj_ntv"   "tx_tfr_val_adj_usd"
## [34] "tx_tfr_val_mean_ntv" "tx_tfr_val_mean_usd" "tx_tfr_val_med_ntv"
## [37] "tx_tfr_val_med_usd"   "tx_tfr_val_ntv"       "tx_tfr_val_usd"
## [40] "vty_day_ret180d"      "vty_day_ret30d"       "vty_day_ret60d"
## [43] "date_time_utc"
```

Now that I imported the `PredictCrypto` package and the data is in snake_case, I can use the function `calculate_percent_change()` to create the target variable to predict. Before I can do that however, I need one more adjustment to the date/time fields, so let's do that using the `anytime`(Eddelbuettel, 2020) package:

```
library(anytime)
googlesheets_data$date <- anytime(googlesheets_data$date)
googlesheets_data$date_time_utc <- anytime(googlesheets_data$date_time_utc)
```

Now I can use the function `calculate_percent_change()` to calculate the % change of the price of each cryptocurrency and add a new column ***target_percent_change*** to each row, which will represent the percentage change in price for the 7 day period that came after that data point was collected:

```
exercise_data <- PredictCrypto::calculate_percent_change(googlesheets_data, 7, 'days')
```

Let's take a peek at the new field:

```
tail(exercise_data$target_percent_change, 10)
```

```
##  [1] -21.577214 -22.986805 -14.995782  -3.220387  -8.205419 -16.015721
##  [7] -16.911053 -12.801616 -16.444769 -17.383620
```

I could easily change this to a 14 day period:

```
calculate_percent_change(googlesheets_data, 14, 'days') %>% tail(10) %>% select(target_percent_ch
```

```
##       target_percent_change
## 10784             -16.13764
## 10785             -13.74175
## 10786             -16.13759
## 10787             -10.74085
## 10788             -17.04613
## 10789             -23.68376
## 10790             -33.13588
## 10791             -31.61660
## 10792             -35.65145
## 10793             -29.77259
```

Or a 24 hour period:

```
calculate_percent_change(googlesheets_data, 24, 'hours') %>% tail(10) %>% select(target_percent_c
```

```
##       target_percent_change
## 10849             -3.215170
## 10850              2.203778
## 10851             -1.520398
## 10852              7.513390
## 10853             -6.282062
## 10854             -5.897307
## 10855            -10.042910
## 10856              1.571641
## 10857             -2.066301
## 10858             -2.626944
```

Disclaimer: **Most of the code to follow was built using the content made available by Allison Hill from the RStudio::conf2020 intro to machine learning workshop and was not code I was familiar with before writing it for this internship application:**

https://education.rstudio.com/blog/2020/02/conf20-intro-ml/

https://conf20-intro-ml.netlify.com/materials/01-predicting/

3.2.2 Feature scaling

```
tic('Feature scaling')
```

Before getting started on the predictive modeling section, it's a good idea for us to scale the numeric data in our dataset. Some of the fields in the dataset are bound to have dramatically different ranges in their values:

```
mean(exercise_data$roi30d, na.rm=T)
```

```
## [1] 20.22271
```

```
mean(exercise_data$cap_mrkt_cur_usd)
```

```
## [1] 17191065374
```

This can be problematic for some models (not every model has this issue), and the difference in the magnitude of the numbers could unfairly influence the model to think that the variable with the larger numbers is more statistically important than the one with the lesser values when that might not actually be true.

For feature scaling, we need to do two things:

1. ***Center*** the data in every column to have a mean of zero

2. ***Scale*** the data in every column to have a standard deviation of one

The `recipes` (Kuhn and Wickham, 2020) package is a very useful package for pre-processing data before doing predictive modeling, and it allows us to center the way we do our data engineering around the independent variable we are looking to predict, which in our case is the `target_percent_change`. We can make a recipe which centers all numeric fields in the data using `step_center()` and then scale them using `step_scale()`. We will also remove the symbol column from the recipe using `step_rm()` because we don't want to use it for the predictions but we don't want to remove it from the dataset either:

```
library(recipes)
scaling_recipe <- recipe(target_percent_change ~ ., data = exercise_data) %>%
  step_center(all_numeric()) %>%
  step_scale(all_numeric())
```

Commented out `step_novel(all_nominal())`, `step_dummy(all_nominal())`, `step_nz(all_predictors())` because size too large and won't run on PC or GitHub Actions.

Now that we have made a data pre-processing *recipe*, let's map it to the `exercise_data` dataset:

```
crypto_data_scaled <- scaling_recipe %>% prep(exercise_data)
crypto_data_scaled
```

```
## Data Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         46
##
## Training data contained 10828 data points and 3120 incomplete rows.
##
## Operations:
##
## Centering for adr_act_cnt, blk_cnt, ... [trained]
## Scaling for adr_act_cnt, blk_cnt, ... [trained]
```

Now let's use `bake()` to put the old dataset in the oven and get back the scaled data :

```
crypto_data_scaled <- crypto_data_scaled %>% bake(exercise_data)
```

Now the values are scaled:

```
head(crypto_data_scaled$cap_mrkt_cur_usd,5)
```

```
## [1] -0.4306519 -0.4306511 -0.4306502 -0.4310321 -0.4304881
```

You can see the difference from the previous values:

```
head(exercise_data$cap_mrkt_cur_usd,5)
```

```
## [1] 86768713 86801326 86834707 71666978 93274716
```

```
toc()
```

```
## Feature scaling: 0.17 sec elapsed
```

3.2.3 Predictive Modeling

```
tic('Predictive Modeling')
```

We can create models using `parsnip` (Kuhn and Vaughan, 2020), which is particularly nice because it gives a very standardized structure for a variety of models. Here's the slightly over-complicated `lm()` linear regression model using `parsnip`:

```
library(parsnip)
lm_model <- linear_reg() %>%
  set_engine("lm") %>%
  set_mode("regression")
```

List of models to refer to: https://tidymodels.github.io/parsnip/articles/articles/Models.html

Random Forest:

```
random_forest_model <- rand_forest(trees = 1000) %>%
  set_engine("randomForest") %>%
  set_mode("regression")
```

XGBoost:

```
xgboost_model <- xgboost_parsnip <- boost_tree(trees=1000) %>%
  set_engine("xgboost") %>%
  set_mode("regression")
```

Remove the fields we will not be be using for the predictive modeling:

```
exercise_data <- exercise_data %>% select(-date_time_utc, -date_time, -pkDummy, -pkey, -cap_real_
```

Before we can start fitting a predictive model, we need to create a train/test split, we can use `rsample`(Kuhn et al., 2019) to put 80% of the data into `crypto_train` and 20% of the data in `crypto_test`:

```
library(rsample)

set.seed(250)
crypto_data <- initial_split(exercise_data, prop = 0.8)
crypto_train <- training(crypto_data)
crypto_test  <-  testing(crypto_data)
```

Compare the number of rows:

```
nrow(crypto_train) # 80% of rows
```

```
## [1] 8663
```

```
nrow(crypto_test) # 20% of rows
```

```
## [1] 2165
```

3.2.4 Fit the model:

Now we can go ahead and train/fit the models to the data:

```
library(modelr)
lm_fitted <- lm_model %>% fit(target_percent_change ~ ., data=crypto_train)
```

Random Forest:

```
tic('Random Forest')
```

```r
random_forest_fitted <- random_forest_model %>%
  fit(target_percent_change ~ ., data = crypto_train)
```

```r
toc()
```

```
## Random Forest: 141.08 sec elapsed
```

XGBoost:

```r
tic('XGBoost')
```

```r
xgboost_fitted <- xgboost_model %>% fit(price_usd ~ ., data=crypto_train)
```

```r
toc()
```

```
## XGBoost: 19.14 sec elapsed
```

Use the trained model to make predictions on test data:

```r
library(tidymodels)
```

```r
lm_predictions <- predict(lm_fitted, crypto_test)
```

```r
xgboost_predictions <- xgboost_fitted %>% predict(crypto_test)
```

Join the full dataset back to the predictions:

```r
lm_predictions <- lm_predictions %>% bind_cols(crypto_test)

xgboost_predictions <- xgboost_predictions %>% bind_cols(crypto_test)
```

Get metrics:

```r
lm_predictions %>%
  metrics(truth = target_percent_change, estimate = .pred)
```

```
## # A tibble: 3 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard     16.9
## 2 rsq     standard      0.0357
## 3 mae     standard     11.0
```

```
xgboost_predictions %>%
  metrics(truth = target_percent_change, estimate = .pred)
```

```
## # A tibble: 3 x 3
##   .metric .estimator  .estimate
##   <chr>   <chr>           <dbl>
## 1 rmse    standard   2463.
## 2 rsq     standard      0.00299
## 3 mae     standard    875.
```

3.2.5 Now make one model for each cryptocurrency.

Lots of code adapted from: https://r4ds.had.co.nz/many-models.html

First I group the data by the cryptocurrency symbol:

```
crypto_data_grouped <- exercise_data %>% group_by(symbol) %>% nest()
```

```
crypto_data_grouped
```

```
## # A tibble: 5 x 2
## # Groups:   symbol [5]
##   symbol data
##   <chr>  <list>
## 1 ETH    <tibble [1,660 x 40]>
## 2 BTC    <tibble [3,507 x 40]>
## 3 LTC    <tibble [2,519 x 40]>
## 4 DASH   <tibble [2,206 x 40]>
## 5 BCH    <tibble [936 x 40]>
```

Make a helper function with the model so I can make the `lm()` model to apply to each cryptocurrency using `purrr`:

```
grouped_linear_model <- function(df) {
  lm(target_percent_change ~ ., data = df)
}
```

I could have made a more complex model here, but decided to keep things a bit simpler with linear regression

Now we can use `purrr`(Henry and Wickham, 2019) to apply the model to each element of the grouped dataframe:

```
grouped_models <- map(crypto_data_grouped$data, grouped_linear_model)
```

The models can be added into the dataframe as nested lists. We can also add the corresponding residuals:

```
crypto_data_grouped <- crypto_data_grouped %>%
  mutate(model=map(data,grouped_linear_model)) %>%
  mutate(resids = map2(data, model, add_residuals))
```

Let's look at the object again:

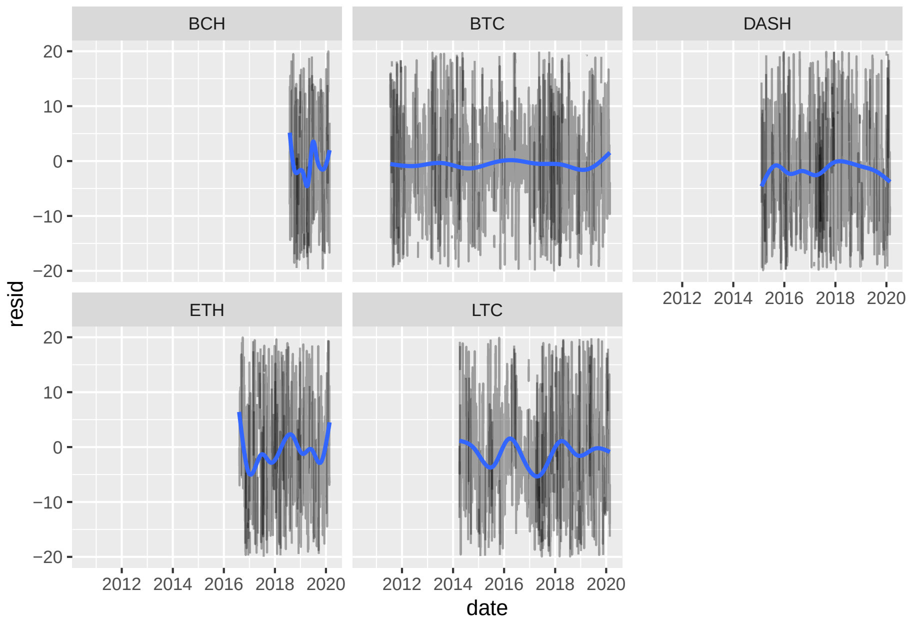
```
crypto_data_grouped
```

```
## # A tibble: 5 x 4
## # Groups:   symbol [5]
##   symbol data                  model  resids
##   <chr>  <list>                <list> <list>
## 1 ETH    <tibble [1,660 x 40]> <lm>   <tibble [1,660 x 41]>
## 2 BTC    <tibble [3,507 x 40]> <lm>   <tibble [3,507 x 41]>
## 3 LTC    <tibble [2,519 x 40]> <lm>   <tibble [2,519 x 41]>
## 4 DASH   <tibble [2,206 x 40]> <lm>   <tibble [2,206 x 41]>
## 5 BCH    <tibble [936 x 40]>   <lm>   <tibble [936 x 41]>
```

Let's unnest the residuals to take a closer look:

```
resids <- unnest(crypto_data_grouped, resids)
```

```
resids %>%
  ggplot(aes(date, resid)) +
    geom_line(aes(group = symbol), alpha = 1 / 3) +
    geom_smooth(se = FALSE) +
    ylim(c(-20,20)) +
    facet_wrap(~symbol)
```



3.2.6 Add Metrics

Now we can use `broom` (Robinson and Hayes, 2020) to get all sorts of metrics back on the models:

```
library(broom)
crypto_models_metrics <- crypto_data_grouped %>% mutate(metrics=map(model,broom::glance)) %>% unn
```

Sort the new tibble by the best r squared values:

```
crypto_models_metrics %>% arrange(-r.squared)
```

```
## # A tibble: 5 x 15
## # Groups:   symbol [5]
##   symbol data  model resids r.squared adj.r.squared sigma statistic  p.value
##   <chr>  <lis> <lis> <list>     <dbl>         <dbl> <dbl>     <dbl>    <dbl>
## 1 BCH    <tib~ <lm>  <tibb~     0.478         0.442  16.1      13.6 1.61e-54
## 2 ETH    <tib~ <lm>  <tibb~     0.300         0.279  15.1      14.5 3.62e-73
## 3 DASH   <tib~ <lm>  <tibb~     0.213         0.196  15.5      12.5 1.95e-68
## 4 LTC    <tib~ <lm>  <tibb~     0.193         0.179  16.8      13.7 3.99e-74
## 5 BTC    <tib~ <lm>  <tibb~     0.151         0.142  12.5      15.4 1.51e-85
## # ... with 6 more variables: df <int>, logLik <dbl>, AIC <dbl>, BIC <dbl>,
## #   deviance <dbl>, df.residual <int>
```

```
toc()
```

```
## Predictive Modeling: 161.54 sec elapsed
```

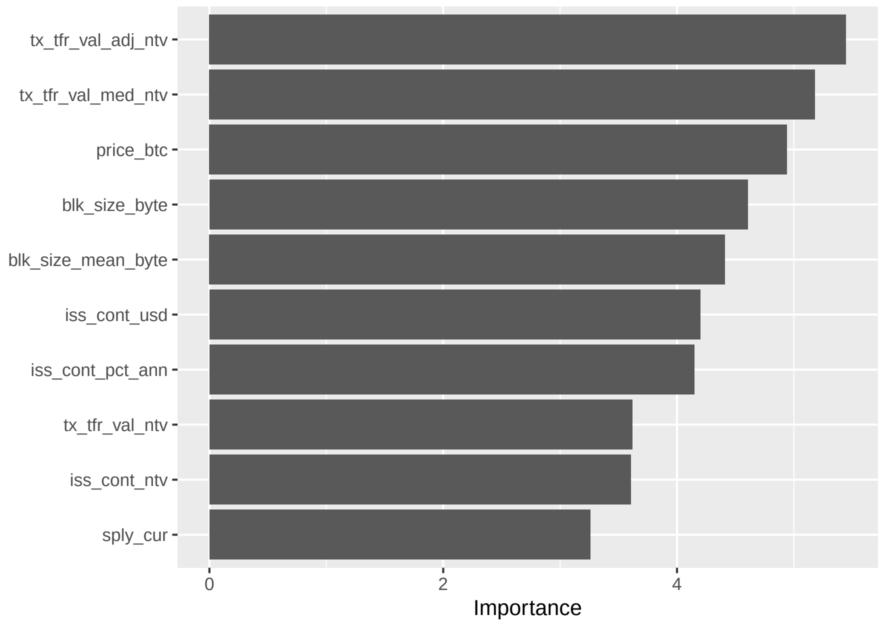
3.2.7 Plot Variable Importance

Now I can use the `vip` (Greenwell et al., 2020) package to plot the variable importance:

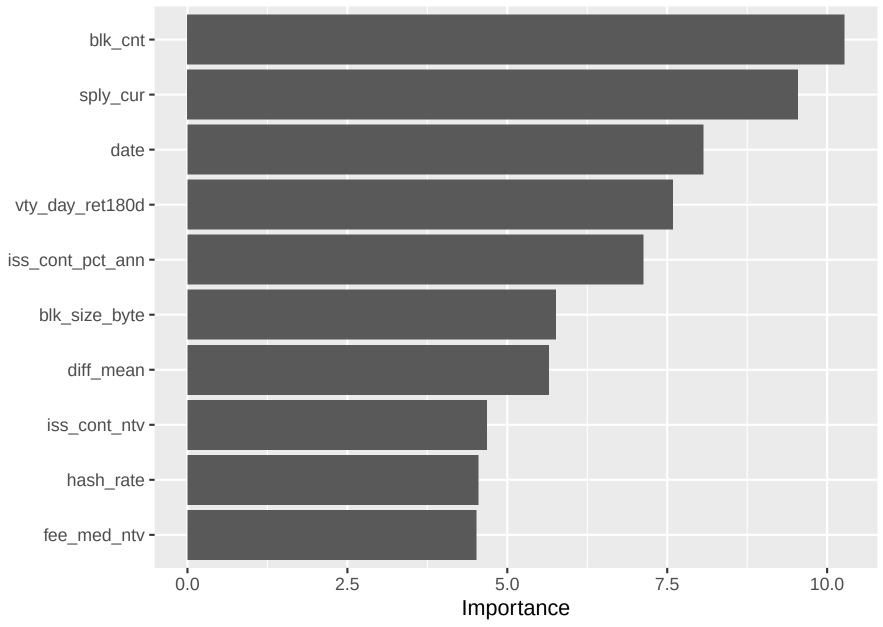
```
library(vip)
```

```
library(vip)
for (i in 1:length(crypto_models_metrics$symbol)){
  print(paste("Now showing", crypto_models_metrics$symbol[[i]], "variable importance:")
  print(vip(crypto_models_metrics$model[[i]]))
}
```

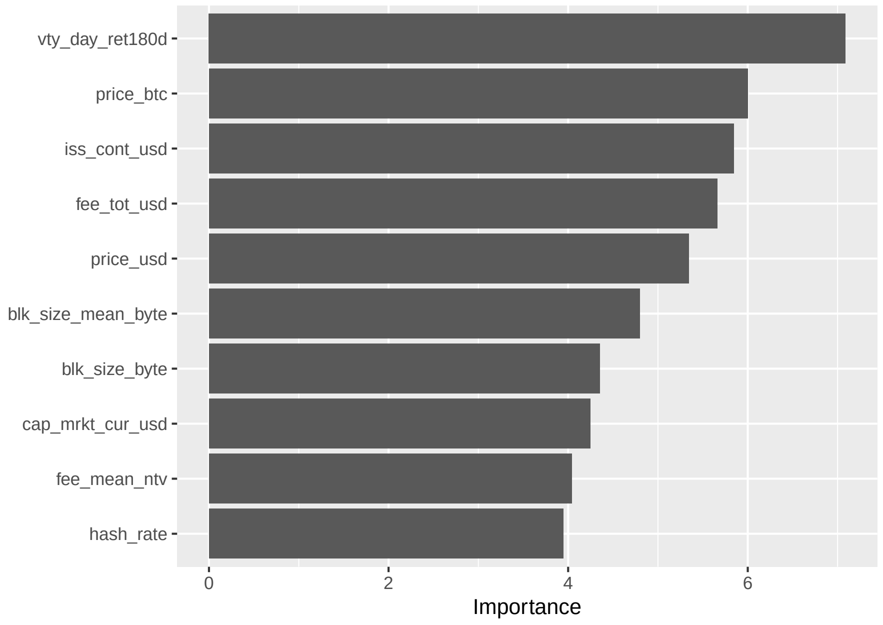
```
## [1] "Now showing ETH variable importance:"
```



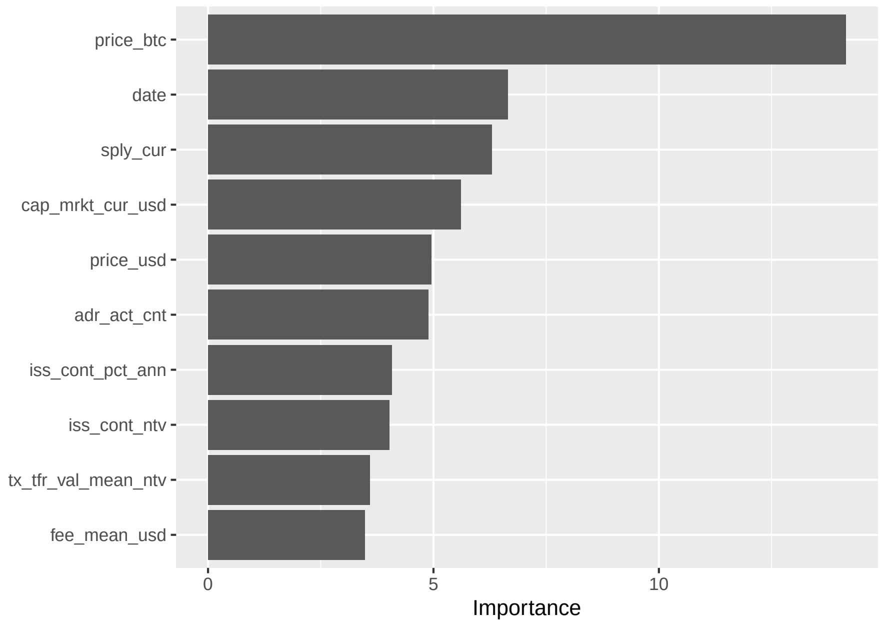
```
## [1] "Now showing BTC variable importance:"
```



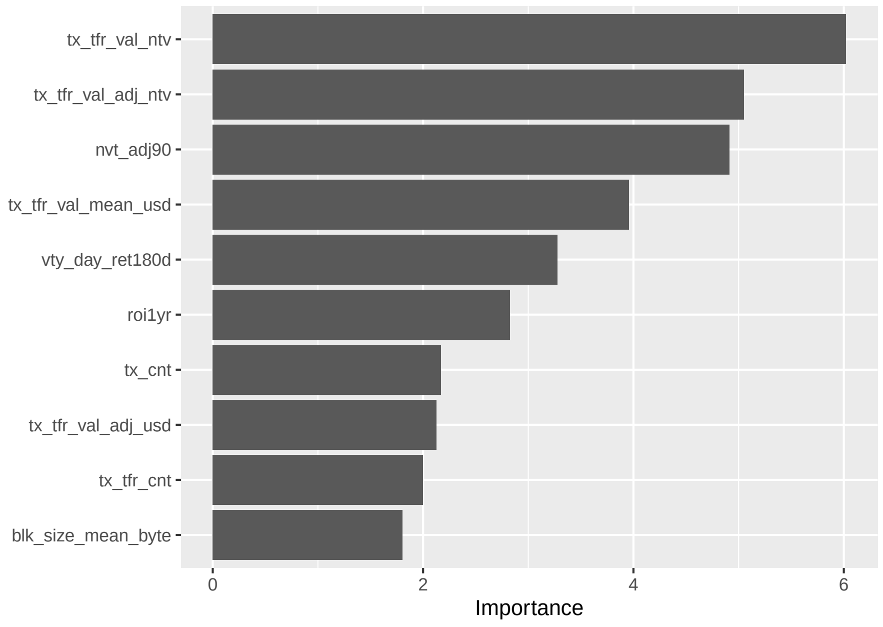
```
## [1] "Now showing LTC variable importance:"
```



```
## [1] "Now showing DASH variable importance:"
```



```
## [1] "Now showing BCH variable importance:"
```



3.2.8 If I were to keep going...

Here are some of the next steps I would take if I were to keep going with this analysis:

1. How much better do the models get if we add timeseries components like Moving Averages?
2. Use parsnip + purrr to iterate through lots of predictive models rather than just applying a simple `lm()` model to each.
3. How much better do the models get with hyperparameter tuning? I would use dials since it's a part of tidymodels.
4. Visualize the best model before and after parameter tuning and then do the same with the worst performing model.
5. I would also go back to the train/test split and use 10-fold cross validation instead.

3.3 Build interactive learnr tutorials for Python using reticulate

I think I could be a great fit for the third project listed related to creating learnr tutorials for Python using reticulate. I have a fair amount of experience in Python, but it's never really clicked very much for as much as R in the past, and I am looking to step-up my Python skills. My Master's in Data Science will work with Python a lot, and people immediately ask if I make tutorials in Python when I show them the R tutorials I have made, so this would be a great one for me to work on. I am also constantly told that Python is better than R for the incorrect reasons, and being more of an expert in Python would certainly help me debunk that myth when someone makes that argument.

I am very familiar with the `reticulate` package and I have used it in the past in an RMarkdown file to make automated cryptocurrency trades through a Python package `shrimpy-python`, which worked really well: https://github.com/shrimpy-dev/shrimpy-python

Since I have already demonstrated my familiarity with learnr tutorials **in the previous section**, I did not make a very extensive example here, but instead created a learnr tutorial with a Python code chunk instead:

Please Wait

Return the total runtime of all of the examples above:

```
toc()
```

```
## Total section 3 runtime: 393.47 sec elapsed
```

Chapter 4

About Me

My formal education is more oriented towards business, but during my time in college I tried to focus on learning tangible skills as much as possible, and computer science/programming started becoming more my niche over time vs. business. I am working towards a Master's in Data Science at the University of Denver and I live in Boulder, Colorado.

I was born and raised in Milan, Italy and moved to the middle of Manhattan with my American mom when I was 16 where I finished highschool. I ended up coming to Boulder after reading a book about ultra marathon running and I could really see myself living in Boulder running with a husky dog training for ultra marathons, so that's what I did:

I also have an adorable little ferret:

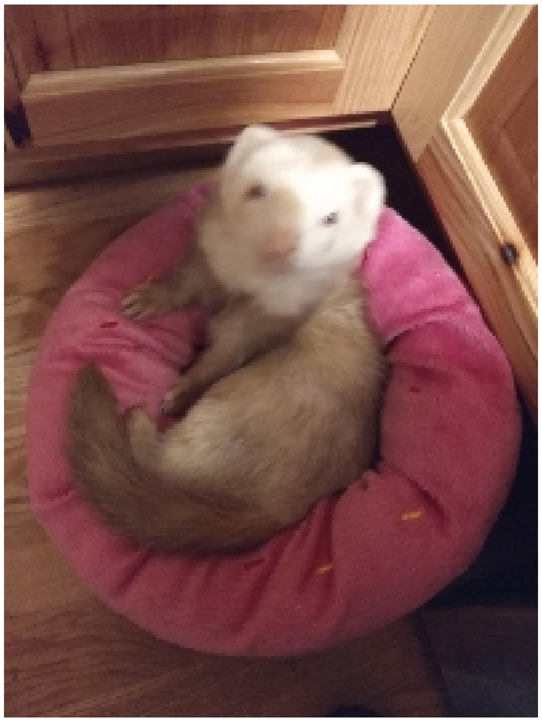


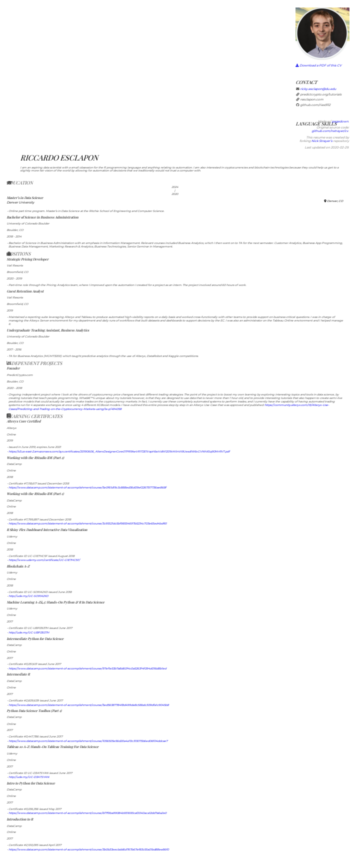
Figure 4.1: Percy

I spent my time as an undergrad at the University of Colorado, Boulder studying information management and being a part of the CU Triathlon club team. As a Junior at CU I was one of the TA's for MGMT 3200 (Business Analytics) after ***really*** enjoying the course, which used Alteryx for doing ETL work, and a tool called DataRobot for making predictions. The grade obtained by students in the class was strongly impacted by their team's percentile ranking on the leaderboards of real Kaggle competitions that were going on at the time, and I found myself really loving working with data and I have been trying to learn as much as I can about anything and everything relating to data science since then. I spent multiple summers locking myself 40+ hours a week in my office working on just that.

Here are some of the online courses I worked my way through:

- https://resclapon.com/datacamp-certificates
- https://resclapon.com/udemy-certificates

Here is my resume with the same online learning certificates added to it:



Download a PDF of this CV

CONTACT

ricky.esclapon@du.edu
predictcrypto.org/tutorials
resclapon.com
github.com/ries9112

LANGUAGE SKILLS

Made w/ pagedown.
Original source code:
github.com/nstrayer/cv.

This resume was created by forking Nick Strayer's repository

Last updated on 2020-02-29.

RICCARDO ESCLAPON

I am an aspiring data scientist with a small obsession for the R programming language and anything relating to automation. I am also interested in cryptocurrencies and blockchain technologies because they could help us get to a slightly more fair vision of the world by allowing for automation of decisions that we traditionally would not trust a computer with.

UCATION

2024
/
2020

Master's in Data Science
Denver University
Denver, CO

- Online part time program. Master's in Data Science at the Ritchie School of Engineering and Computer Science.

Bachelor of Science in Business Administration

University of Colorado Boulder

Boulder, CO

2018 - 2014

- Bachelor of Science in Business Administration with an emphasis in Information Management. Relevant courses included Business Analytics, which I then went on to TA for the next semester. Customer Analytics, Business App Programming, Business Data Management, Marketing Research & Analytics, Business Technologies, Senior Seminar in Management.

OSITIONS

Strategic Pricing Developer

Vail Resorts

Broomfield, CO

2020 - 2019

- Part-time role through the Pricing Analytics team, where I improved upon the automation I created for a project as an intern. The project involved around 60 hours of work.

Guest Retention Analyst

Vail Resorts

Broomfield, CO

2019

- Maintained a reporting suite leveraging Alteryx and Tableau to produce automated high-visibility reports on a daily basis and worked cross-functionally with other teams to provide support on business critical questions. I was in charge of managing the Alteryx Server environment for my department and daily runs of workflows that built datasets and dashboards to support decisions by the EC. I was also an administrator on the Tableau Online environment and I helped manage it.

Undergraduate Teaching Assistant, Business Analytics

University of Colorado Boulder

Boulder, CO

2017 - 2016

- TA for Business Analytics (MGMT3200) which taught predictive analytics through the use of Alteryx, DataRobot and Kaggle competitions.

DEPENDENT PROJECTS

Founder

PredictCrypto.com

Boulder, CO

2020 - 2018

- Ongoing independent project where I look at the drivers of cryptocurrency price changes and test different trading strategies. The point of the project is to boost my own learning by exploring interesting topics and tools in data science, by creating tutorials that teach people programming. Whatâ€™s unique about my tutorials however, is that they are designed to use data that is never more than 1 hour old, and to provide interesting tutorials that also happen to create semi-live predictive models that could actually be used to make trades on the cryptocurrency markets. In fact, I currently use these completely automated systems to perform trades, and I currently have the capability of having automated trading systems run for 4 separate exchanges at once using 4 different XGBoost models. I have outlined this entire process step by step in an Alteryx Use Case that was approved and published: https://community.alteryx.com/t5/Alteryx-Use-Cases/Predicting-and-Trading-on-the-Cryptocurrency-Markets-using/ta-p/494058

EARNING CERTIFICATES

Alteryx Core Certified

Alteryx

Online

2019

- Issued in June 2019, expires June 2021
- https://s3.us-east-2.amazonaws.com/ayx.certificates/20190606_AlterxDesignerCore0TPR9keVR1T2E7VapHboVdW1201kMiInVt9UesdfW6xGVNhXSq90MrRVT.pdf

Working with the RStudio IDE (Part 2)

DataCamp

Online

2018

- Certificate #7,118,617 issued December 2018
- https://www.datacamp.com/statement-of-accomplishment/course/5e0f61d19c3c888ed36d09ef2267517736ae868f

Working with the RStudio IDE (Part 1)

DataCamp

Online

2018

- Certificate #7,799,897 issued December 2018
- https://www.datacamp.com/statement-of-accomplishment/course/3c95521dc5bf98554697b5234c703e65ed4bdf61

R Shiny Flex Dashboard Interactive Data Visualization

Udemy

Online

2018

- Certificate ID UC-G1E7HCSF issued August 2018
- https://www.udemy.com/certificate/UC-G1E7HCSF/

Blockchain A-Z

Udemy

Online

2018

- Certificate ID UC-SO91A2XD issued June 2018
- http://ude.my/UC-SO91A2XD

Machine Learning A-Zâ„¢: Hands-On Python & R In Data Science

Udemy

Online

2017

- Certificate ID UC-UBPZ637H issued June 2017
- http://ude.my/UC-UBPZ637H

Intermediate Python for Data Science

DataCamp

Online

2017

- Certificate #3,051,631 issued June 2017
- https://www.datacamp.com/statement-of-accomplishment/course/97ef1e53b7a8d60f4c0a5263f4f094d016d8b1ed

Intermediate R

DataCamp

Online

2017

- Certificate #2,609,639 issued June 2017
- https://www.datacamp.com/statement-of-accomplishment/course/5ed56387784f8d499de8c58bdc509dfa1c9045b8

Python Data Science Toolbox (Part 1)

DataCamp

Online

2017

- Certificate #3,447,786 issued June 2017
- https://www.datacamp.com/statement-of-accomplishment/course/1056505e56d20a4a72c3130173b6ed06f04ddcae7

Tableau 10 A-Z: Hands-On Tableau Training For Data Science

Udemy

Online

2017

- Certificate ID UC-D3ATSYAN issued June 2017
- http://ude.my/UC-D3ATSYAN

Intro to Python for Data Science

DataCamp

Online

2017

- Certificate #3,036,356 issued May 2017
- https://www.datacamp.com/statement-of-accomplishment/course/6f7f9ba99084b931693caf2040aca12dd7a6a2e0

Introduction to R

DataCamp

Online

2017

- Certificate #2,100,095 issued April 2017
- https://www.datacamp.com/statement-of-accomplishment/course/3b0b33eecbdd6d767b67ef83c55a01bd88ee86f0

Thanks again to Nick Strayer for the awesome template

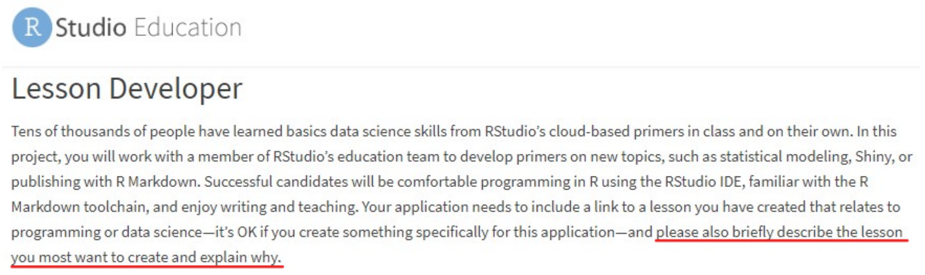
In my senior year at CU Boulder, I did a business analytics internship with the Pricing Analytics team at Vail Resorts, which turned out to be a lot of manual work that should have been automated, so that's what I did. After graduating from CU Boulder I spent some time applying the things I learned around automation and web scraping to setup a project to collect data relating to the cryptocurrency markets from several sources because I had been trading on them since early 2014 and I saw an opportunity for some automated trading, and I wanted to have a personal project that I could work on over a longer period of time and use to get more comfortable in tools like SQL, R and Python with the prospect of making some money in the end. After a year and a half of working on this, I realized that I was actually better off opening up my project to others more and using it as a tool to teach others to program and that could be really valuable in my career progression towards eventually being a competent data scientist rather than spending all of my energy trying to make short-term trades, which doesn't really teach any tangible or useful skills outside of its own domain. The tutorials and lessons I am building specifically around this project can be found on https://predictcrypto.org/.

Today I am working towards a master's in Data Science part-time online and working on creating more research (including a more legitimate research paper with two professors) around the *PredictCrypto* project, and working on putting out more tutorials and content through a YouTube channel. My master's program allows me to take one course at a time, be a full-time student, or do anything in between, so that flexibility allows me to work full-time without having an overwhelming schedule.

Chapter 5

Ideal Tutorial

5.1 Overview


RStudio Education

Lesson Developer

Tens of thousands of people have learned basics data science skills from RStudio's cloud-based primers in class and on their own. In this project, you will work with a member of RStudio's education team to develop primers on new topics, such as statistical modeling, Shiny, or publishing with R Markdown. Successful candidates will be comfortable programming in R using the RStudio IDE, familiar with the R Markdown toolchain, and enjoy writing and teaching. Your application needs to include a link to a lesson you have created that relates to programming or data science—it's OK if you create something specifically for this application—and please also briefly describe the lesson you most want to create and explain why.

I don't see this question on the last post regarding the RStudio internship applications being open through March 6th, but I have been thinking about this question since I saw it originally posted here in November, so I wanted to include this answer in my application.

Ultimately I think analyzing cryptocurrencies is fun and interesting and a good way to get people's attention, but I have been thinking that a much more useful application of these ideas would be to be able to do a very similar thing but to create live data feeds from sensors out in the real world. This data could then be used to provide highly interactive programming tutorials, where the outcomes of the analysis would change based on the most recent data that was collected. To give a practical example of what that could look like, if there was a live data feed of sensors across Australia giving live information around particulates, carbon monoxide, ozone, carbon dioxide as well as other factors like informaton about the wind, etc.. it seems to me that this could empower things like early detection and much better prevention in general through predictive modeling and being able to triangulate the location of fires as they start or to figure out how to spread the limited resources across the different fires, and I just love the idea of the possibility of moving the needle on a problem through programming tutorials rather than working with uninteresting old data. Creating a system that allows people to actually contribute towards solving real problems might be wishful thinking, but I do believe that the best to teach someone these concepts is to give them data they care about and give them a realistic path forward to apply their existing intuition to answer questions they care about. The `mtcars` and `iris` datasets are great, but data feeds that change over time would be better in ***some*** cases in terms of getting a person invested in the actual analysis being done.

5.2 Tutorials I Have Planned

As I was thinking through this question, I realized that beyond doing cool work around the data being used itself, I have a pretty lengthy list of topics that I feel are not always expressed as concisely as they should be. These topics *have* been covered by others in the past, but if I had a 5-10 minute video outlining things the way I plan on doing it, it would have saved me a lot of time, so hopefully even if I only reach a couple of people I will have saved them a lot of time, as well as myself whenever I want to go back to using **any of these tools:**

- Creating a website with bookdown, GitHub and Netlify:
  - Conceptually speaking this is amazingly simple to implement if you just know where to click in GitHub + RStudio and does things that would be pretty difficult to achieve with older tools.
- GitHub actions for automation:
  - This is a pretty new topic because GitHub Actions have been very recently introduced and most resources online make this way over-complicated and/or they are not usually specific to R. It's actually not that difficult though and it makes a ton of sense conceptually, especially when using `devtools::check()`, and is a general tool that can be used for all sorts of automation. In fact, it would be a terrible experience, but you could program in R without needing your own computer by using GitHub actions.
  - I was running into an issue I did not understand when using GitHub actions with bookdown files because of the default argument `clean_envir=FALSE` when running `render_book()`, and I documented the issue here: https://community.rstudio.com/t/github-actions-object-from-secrets-not-found/54519/5
  - After making a video tutorial around making a website with bookdown, I plan on using that project to explain github actions in another video.

- Using blogdown and pagedown
- Making an R package
- Creating tests to go along with an R package:
  - Including code coverage and having the custom badge on the GitHub page refreshed through GitHub actions
- General overview of how to use GitHub with RStudio:
  - In companies you would have a development space and a **production** environment and I see my personal use of GitHub + RStudio as being very similar to that. When you make changes locally it's conceptually similar to a dev environment, and when you push things to GitHub those changes are published to the production environment where it has downstream effects, for example triggering a new build for a website.
- Setting up R + RStudio + GitHub:
  - Downloading R
  - Downloading RStudio
  - Downloading GitHub and pointing the global options within RStudio to point to git.exe to prompt RStudio to ask for login and create the **Git** tab in the IDE
- Flexdashboards
- Web Scraping
- Using RStudio Add-Ins and coolest ones
- Awesome ggplot2 extensions:
  - trelliscope
  - rayshader + rayrender
  - ggmap
  - gganimate
  - gghighlight
  - … LOTS more
- Understanding the tidyverse. Here's a quick example of the things I would really drive home in a tutorial around the tidyverse and when/why you would want to use the pipe operator:

Take the following example:

```
sqrt(25)
```

```
## [1] 5
```

Here it is easy enough to keep track of what is happening. We are taking the square root of 25 and nothing more. But let's say we have a more complex operation:

```
abs(exp(sqrt(25)))
```

```
## [1] 148.4132
```

As the code gets more complicated, it gets more difficult to read the code. What order do the operations run? Things can get pretty out of hand, this is not a particularly extreme example.

In comes the pipe operator! Using the %>% we **start** with the value being manipulated, and apply each operation one step at a time:

```
25 %>%
  sqrt() %>%
  exp() %>%
  abs()
```

```
## [1] 148.4132
```

Now it becomes much clearer that our code starts with the value 25 and the functions are applied in the order `sqrt()`, `exp()`, `abs()`.

When we work with a full dataset, this will also work much better because it will be much easier to distinguish between the data we want to apply a transformation to and the actual transformation. Let's walk through one more example to illustrate this idea.

Let's make a very simple example dataset:

```
data <- data.frame("numbers"=c(3,7,9))
data
```

```
##   numbers
## 1       3
## 2       7
## 3       9
```

Without using the pipe operator, this is what the usage of the `filter()` function would look like:

```
filter(data, numbers > 7)
```

```
##   numbers
## 1       9
```

Treating the object `data` within the `filter()` function is not clear. Using the pipe operator, this operation becomes more clear:

```
data %>% filter(numbers > 7)
```

```
##   numbers
## 1       9
```

To make this point clear, try to translate this code to english in your head:

```
round(log(sqrt(filter(data, numbers > 7))),3)
```

```
##   numbers
## 1   1.099
```

Not exactly straightforward right? Now try to translate this code in your head and see if it is easier at all:

```
data %>%
  filter(numbers > 7) %>%
  sqrt() %>%
  log() %>%
  round(3)
```

```
##   numbers
## 1   1.099
```

You could read this line by line as:

1. Start with the dataframe object called `data`
2. Filter the rows based on the column called `numbers` having a value larger than 7
3. Take the square root of the result
4. Take the log of the result
5. Round the result by 3 decimal places

5.3 Final notes

I would also have a version of each planned tutorial recorded in Italian, because I am bilingual and most of this content does not currently exist in Italian as far as I can tell.

Chapter 6

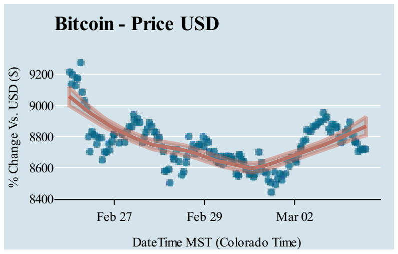
Cool Charts

Some examples of using packages like `plotly` (Sievert et al., 2019) and `rayshader` (Morgan-Wall, 2020) to

```
tic('Cool charts section')
```

6.1 Price USD - Last 7 Days - BTC

```
plot <- ggplot(subset(cryptoData, Name == 'Bitcoin'), aes(x = DateTimeUTC, y = PriceUSD)) +

  geom_jitter(alpha=0.75, colour='deepskyblue4') +

  labs(title=paste('Bitcoin', ' - Price USD'),
       x='DateTime MST (Colorado Time)', y='% Change Vs. USD ($)') +

  geom_smooth(colour='coral3')  + theme_economist()

ggplotly(plot)
```



6.2 Rayshader Hex Bins - Previous 24h % Change - Whole Market

First some camera settings for the rayshader video:

```r
phivechalf = 30 + 60 * 1/(1 + exp(seq(-7, 20, length.out = 180)/2))
phivecfull = c(phivechalf, rev(phivechalf))
thetavec = -90 + 45 * sin(seq(0,359,length.out = 360) * pi/180)
zoomvec = 0.45 + 0.2 * 1/(1 + exp(seq(-5, 20, length.out = 180)))
zoomvecfull = c(zoomvec, rev(zoomvec))
filename_movie = 'rayshader.mp4'
```

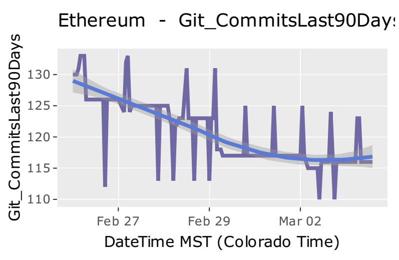
```r
plot_gg(ggplot(data = cryptoData, aes(x=DateTimeUTC, y=PercChange24hVsUSD)) +
          geom_hex() +
          ylim(-20,20))

render_movie(filename = filename_movie, type = "custom",
             frames = 360,  phi = phivecfull, zoom = zoomvecfull, theta = thetavec)
```

```
## [1] "C:\\Users\\Ricky\\Desktop\\RStudio-Internship-Application\\rayshader.mp4"
```

```
rgl::rgl.close()
```

6.3 Git Commits Last 90 Days - ETH

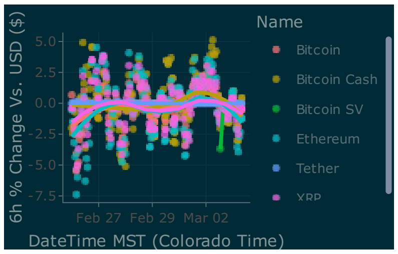
```
plot <- ggplot(filter(cryptoData, Name == 'Ethereum'), aes_string(x = "DateTimeUTC", y = 'Git_Com
  geom_line(size = 1, alpha=0.75, colour='darkslateblue') +
  labs(title=paste('Ethereum', ' - ', 'Git_CommitsLast90Days'),
       x='DateTime MST (Colorado Time)') +
  geom_smooth()
ggplotly(plot)
```



6.4 Percent Change Over Next 6 Hours - Last 7 Days - Top 5 Ranked

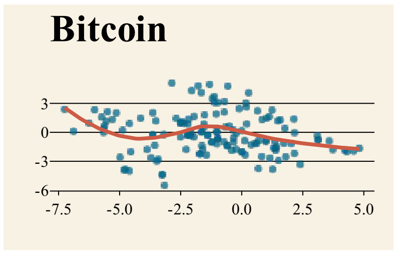
This chart plots the % change for the 6 hour period after it was collected.

```
plot <- ggplot(filter(targetData, Rank >= 1, Rank <= 5 ), aes(x = DateTimeUTC, y = Targ
  geom_jitter(alpha=0.75) +
  labs(x='DateTime MST (Colorado Time)', y='6h % Change Vs. USD ($)') +
  geom_smooth(aes(group=Name), se=F)  + theme_solarized(light=F)
ggplotly(plot)
```



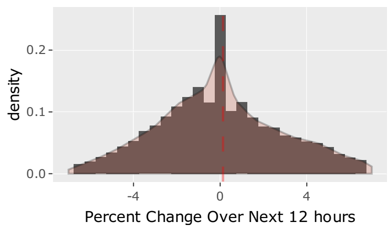
6.5 Plotting the relationship between previous 24h % change and next 24h % change - Bitcoin

```
plot <- ggplot(subset(targetData24, Name == 'Bitcoin'), aes(x = PercChange24hVsUSD, y = TargetPer

  geom_jitter(alpha=0.75, colour='deepskyblue4') +

  labs(title='Bitcoin',
       x='% Change Previous 24h vs. $', y='% Change Next 24h vs. $') +

  geom_smooth(colour='coral3',se=F)  + theme_wsj()

ggplotly(plot)
```

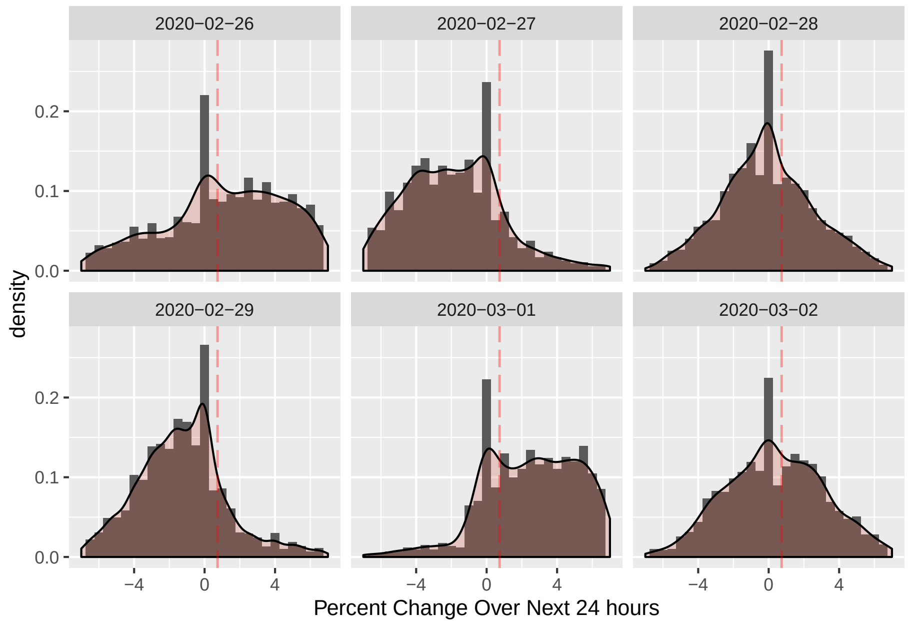


```
# output$pricePlotStats <- renderPlot({
#   ggscatterstats(data = filter(targetData24, Name == input$cryptocurrencyNameRel), x = PercChar
# })
```

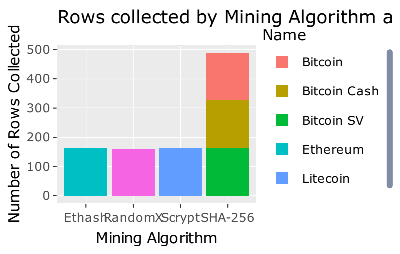
6.6 Percent Change Over Next 12 Hours - Last 7 Days - Top 100 Ranked



6.7 Percent Change Over Next 24 Hours - By Date



6.8 Rows collected by Mining Algorithm - Top 15 Ranked



```
## [1] TRUE
```

```
toc()
```

```
## Cool charts section: 93.77 sec elapsed
```

Bibliography

Bryan, J. (2020). *googlesheets4: Access Google Sheets using the Sheets API V4*. R package version 0.1.0.9000.

Eddelbuettel, D. (2020). *anytime: Anything to 'POSIXct' or 'Date' Converter*. R package version 0.3.7.

Firke, S. (2020). *janitor: Simple Tools for Examining and Cleaning Dirty Data*. R package version 1.2.1.

Gans, M. (2020). *Javascript For Data Science*. Chapman and Hall/CRC, Boca Raton, Florida, 1st edition. ISBN 978-0-367-42248-6.

Greenwell, B., Boehmke, B., and Gray, B. (2020). *vip: Variable Importance Plots*. R package version 0.2.1.

Henry, L. and Wickham, H. (2019). *purrr: Functional Programming Tools*. R package version 0.3.3.

Iannone, R., Allaire, J., and Borges, B. (2018). *flexdashboard: R Markdown Format for Flexible Dashboards*. R package version 0.5.1.1.

Izrailev, S. (2014). *tictoc: Functions for timing R scripts, as well as implementations of Stack and List structures.* R package version 1.0.

Kuhn, M., Chow, F., and Wickham, H. (2019). *rsample: General Resampling Infrastructure*. R package version 0.0.5.

Kuhn, M. and Vaughan, D. (2020). *parsnip: A Common API to Modeling and Analysis Functions*. R package version 0.0.5.

Kuhn, M. and Wickham, H. (2020). *recipes: Preprocessing Tools to Create Design Matrices*. R package version 0.1.9.

Morgan-Wall, T. (2020). *rayshader: Create Maps and Visualize Data in 2D and 3D*. R package version 0.14.1.

Robinson, D. and Hayes, A. (2020). *broom: Convert Statistical Analysis Objects into Tidy Tibbles*. R package version 0.5.4.

Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M., and Despouy, P. (2019). *plotly: Create Interactive Web Graphics via 'plotly.js'*. R package version 4.9.1.

Wickham, H. (2019). *tidyverse: Easily Install and Load the 'Tidyverse'*. R package version 1.3.0.

Xie, Y. (2019). *blogdown: Create Blogs and Websites with R Markdown*. R package version 0.17.

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.17.

Xie, Y., Cheng, J., and Tan, X. (2019). *DT: A Wrapper of the JavaScript Library 'DataTables'*. R package version 0.11.

Xie, Y., Lesur, R., and Thorne, B. (2020). *pagedown: Paginate the HTML Output of R Markdown with CSS for Print*. R package version 0.8.