RStudio 2020 Internship Application

Riccardo Esclapon

Contents

Chapter 1

Overview

Video intro here

https://education.rstudio.com/blog/2020/02/applications-for-2020-intern-program-are-now-open/

APPLICATIONS END ON MARCH 5TH BE SURE TO APPLY BEFORE THEN!!

For video:

Start off with overview of projects I am suited for showing work I did for this application specifically. Then go on to talk about ways I have applied the broad RMarkdown ecosystem and automation in my work. Then talk a bit more about myself. Talk about ideal tutorial overview and close things by mentioning cool charts/visualizations section (outline this at a high level under 2 minutes in the video at the start here)

Chapter 2

What makes me a good fit

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

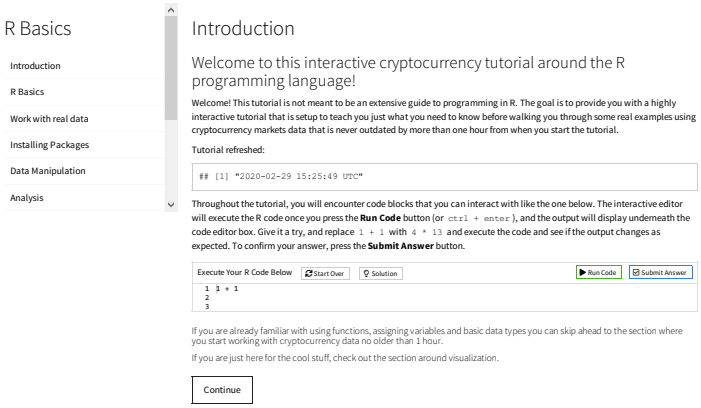
2.1 I .Rmd files

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:

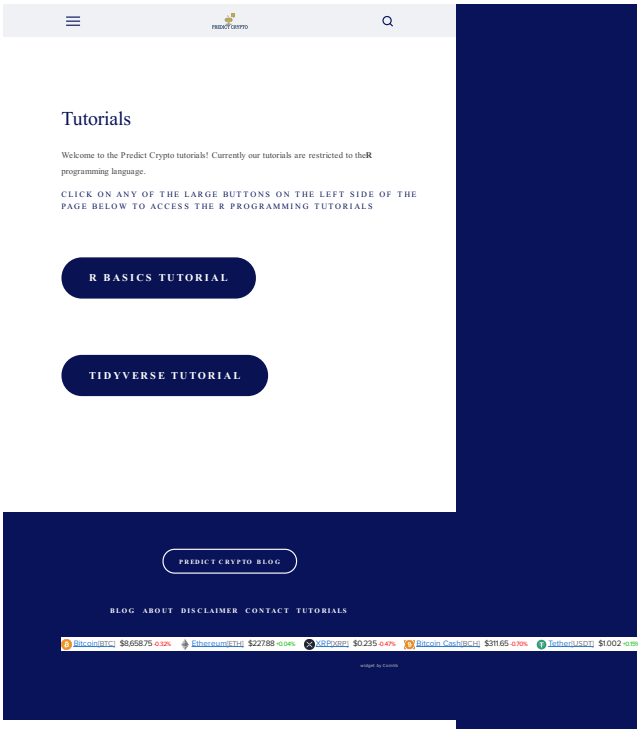
2.1.1 Learnr

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every time the tutorial is opened, users learn to program in R using data from the cryptocurrency markets that is never outdated by more than 1 hour: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)


R Basics
Introduction
R Basics
Work with real data
Installing Packages
Data Manipulation
Analysis
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] "2020-02-29 15:25:49 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.
Execute Your R Code Below
Start Over
Solution
Run Code
Submit Answer
1 1 + 1
2
3
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.
Continue

I would recommend looking at the **Visualization** section to visually see that the data is never outdated by more than 1 hour.



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 (**?**) 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

2020-02-29

1 Overview

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.

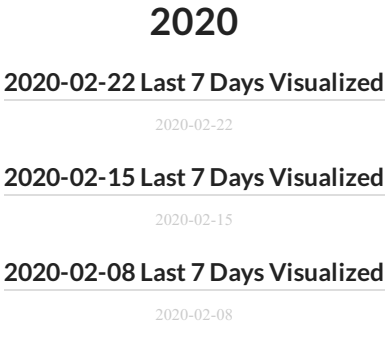
2.1.3 Presentations

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

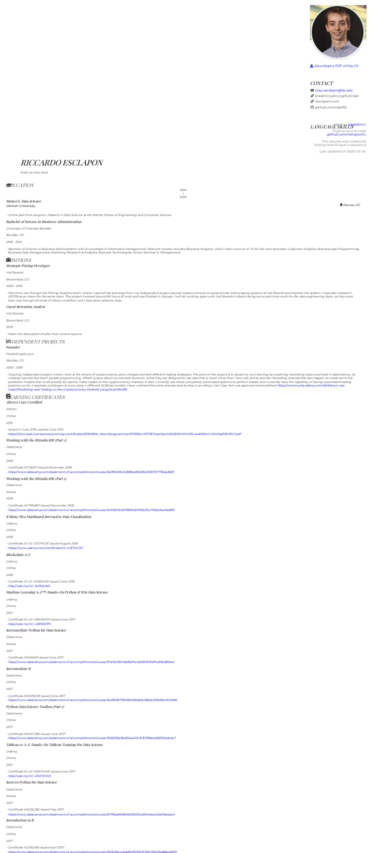
Blogdown(**?**) and bookdown work very similarly, so most of what I mentioned in the bookdown section applies here. Because my website predictcrypto.com only shows the latest data based on the current date, I leverage blogdown to create weekly snapshots of the visualizations over the last 7 day period: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



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(**?**) 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:



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 2020-02-24.

RICCARDO ESCLAPON

Enter an intro here

EDUCATION

Master's, Data Science
Denver University
Denver, CO
2024 / 2020

- 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 used all the learnings from my independent project around cryptocurrencies to setup similar web scraping systems at Vail Resorts and improve the system I originally created in 2017/18 as an intern for the same team. The project involved around 60 hours of work and was finished in January. I will be working again with Vail Resorts in March doing some part-time work for the data engineering team, as they work their way through all kinds of Alteryx workflow and I have been asked to help.

Guest Retention Analyst
Vail Resorts
Broomfield, CO
2019

- Make this description smaller than current resume

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. What'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

LEARNING CERTIFICATES

Alteryx Core Certified
Alteryx
Online
2019

- Issued in June 2019, expires June 2021
- [illegible]

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-G1E7HC5F issued August 2018
- https://www.udemy.com/certificate/UC-G1E7HC5F/

Blockchain A-Z
Udemy
Online
2018

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

Machine Learning A-Z™: 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/5ed563877f84f8d499de8c58bdc309dfa1c9045b8

Python Data Science Toolbox (Part 1)
DataCamp
Online
2017

- Certificate #3,447,786 issued June 2017
- https://www.datacamp.com/statement-of-accomplishment/course/1056505e56d20a4a72c313073b6ed06f04ddcae7

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

2.1.6 Flexdashboard

Flexdashboards(**?**) 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:

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 mindblowing, is that the same methodologies can be applied to make other interfaces, like making a blogdown website, and I can't speak highly enough of all the work Yihui blessed us all with.

2.3 Fit Within the Company

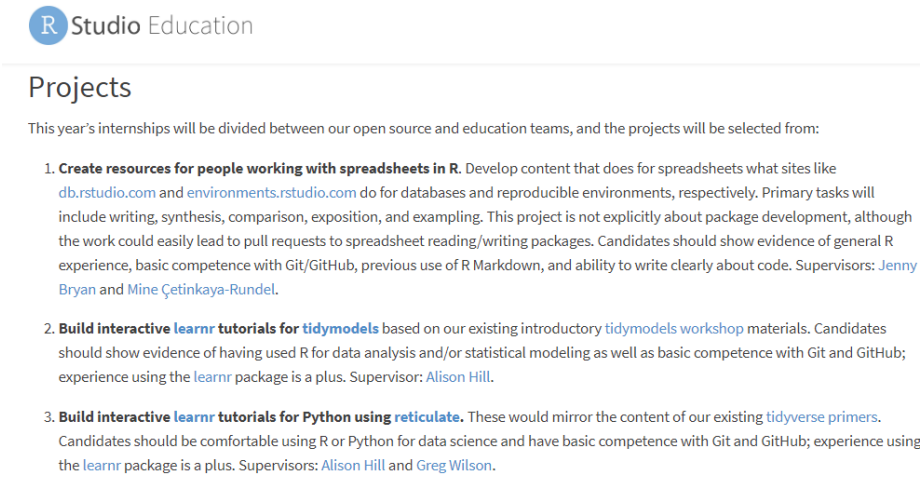
I really wanted to go to RStudio::conf 2019 but was not able to make it out and after all the videos got posted I watched most of them and immediately knew I had to come to RStudio::conf 2020 and it was a truly incredible experience.

JJ's talk and BCorp announcement really resonated with me and there is no other company who's mission I agree with more and I would always do my very best in carrying forward those values. I fundamentally believe the most straightforward way to success is to help other people succeed, and I love the values that RStudio holds dear as a company and there is really no other company that I want to work for more than RStudio.

I also work my best remotely and I have a dedicated office to do my work in with a powerful desktop PC and two screens.

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`(**?**).

First I will go ahead and import every package in the `tidyverse`(**?**):

```
library(tidyverse)
```

Next let's import the `googlesheets4` and read a spreadsheet I made for this

internship application, specifying the sheet called *coinmetrics_btc_eth* inside the function `read_sheet()`:

```
spreadsheet_url <- "https://docs.google.com/spreadsheets/d/1_zRBFrB1au7qhxuDDfDuh_bPLG
library(googlesheets4)
googlesheets_data <- read_sheet(spreadsheet_url, sheet = 'coinmetrics_btc_eth') %>% as
```

```
## Using an auto-discovered, cached token.
## To suppress this message, modify your code or options to clearly consent to the use
## See gargle's "Non-interactive auth" vignette for more details:
## https://gargle.r-lib.org/articles/non-interactive-auth.html
## The googlesheets4 package is using a cached token for ries9112@colorado.edu.
```

Let's take a peek at the `datatable()` using `DT` (**?**)

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

```
## Warning in instance$preRenderHook(instance): It seems your data is too big
## for client-side DataTables. You may consider server-side processing: https://
## rstudio.github.io/DT/server.html
```

Date	Symbol	AdrActCnt	BlkCnt	BlkSizeByte	BlkSizeMeanByte	CapMVRVCur	CapMrktCurUSD
2020-02-24T00:00:00Z	BTC	758122	141	161000000	1141214	1.657654	176000000000
2020-02-24T00:00:00Z	ETH	320273	6481	153000000	23637.33	29000000000	2310000000000000
2020-02-23T00:00:00Z	BTC	622397	142	131000000	924118.8	1.713841	182000000000
2020-02-23T00:00:00Z	ETH	311622	6532	130000000	19886.52	30200000000	2270000000000000
2020-02-22T00:00:00Z	BTC	660812	148	136000000	919925.9	1.663798	176000000000

This data is sourced from the website coinmetrics.io

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

3.1.1 Predictive Model - Data Prep

Using the data from coinmetrics, I will create a predictive model to forecast the percentage change in price over time in order to show that I would be a good fit for the first two projects listed.

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: https://ries9112.github.io/PredictCrypto/

PredictCrypto

codecov 56%

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(?)` 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] "DateTimeUTC"
```

```
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"     "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(?)` 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]   2.371123   7.142857  20.891949  23.846718   6.707045  19.548486
##  [7] -25.610849 -21.561229 -30.693069 -41.121855
```

I could easily change this to a 14 day period:

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

```
##      target_percent_change
## 5104             18.281533
## 5105             25.714286
## 5106             20.496231
## 5107              8.184818
## 5108              1.779083
## 5109              4.327005
## 5110            -28.003738
## 5111            -19.701349
## 5112            -25.742574
## 5113            -28.821062
```

Or a 24 hour period:

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

```
##      target_percent_change
## 5130              0.492989
## 5131              4.682143
## 5132             10.801132
```

```
## 5133             -7.332233
## 5134             -9.989603
## 5135              3.630922
## 5136            -26.167717
## 5137              5.963659
## 5138             -7.504950
## 5139             -5.871389
```

Let's also scale the variables

ADD FEATURE SCALING HERE

3.2 Build interactive learnr tutorials for tidymodels

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

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

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.**

3.2.1 Create the `parsnip` (?) model

```
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 = 100, mode = "regression") %>%
  set_engine("randomForest")
```

XGBoost:

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

Remove fields not used for models (NOTE: REMOVE price_usd_x_daysLater AND date_time_x_days_Later FROM BEING GENERATED INSIDE THE FUNCTION)

```
exercise_data <- select(exercise_data, -date_time_utc, -date_time, -pkDummy, -pkey, -pr
```

Create train/test split using `rsample`(**?**):

(should I do 10-fold cross validation?)

```
library(rsample)

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

3.2.2 Train/fit the model:

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

Random Forest:

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

XGBoost:

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

Use the trained model to make predictions on test data:

```
library(tidymodels)
```

```
## -- Attaching packages ------------------------------------------------------ tidymod
```

```
## v broom     0.5.4     v tune      0.0.1
## v dials     0.0.4     v workflows 0.1.0
## v infer     0.5.1     v yardstick 0.0.5
## v recipes   0.1.9

## -- Conflicts ------------------------------------------------------ tidymodels_conflicts()
## x data.table::between()   masks dplyr::between()
## x broom::bootstrap()      masks modelr::bootstrap()
## x scales::discard()       masks purrr::discard()
## x dplyr::filter()         masks plotly::filter(), stats::filter()
## x data.table::first()     masks dplyr::first()
## x recipes::fixed()        masks stringr::fixed()
## x dplyr::lag()            masks stats::lag()
## x data.table::last()      masks dplyr::last()
## x yardstick::mae()        masks modelr::mae()
## x yardstick::mape()       masks modelr::mape()
## x dials::margin()         masks ggplot2::margin()
## x yardstick::rmse()       masks modelr::rmse()
## x yardstick::spec()       masks readr::spec()
## x recipes::step()         masks stats::step()
## x data.table::transpose() masks purrr::transpose()
## x recipes::yj_trans()     masks scales::yj_trans()
```

```
lm_predictions <- lm_fitted %>% predict(crypto_test)
```

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

Join the full dataset back to the predictions:

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

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

Get metrics:

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

```
## # A tibble: 3 x 3
##   .metric .estimator   .estimate
##   <chr>   <chr>            <dbl>
## 1 rmse    standard   3580.
## 2 rsq     standard      0.000277
## 3 mae     standard   1749.
```

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

```
## # A tibble: 3 x 3
##   .metric .estimator   .estimate
##   <chr>   <chr>            <dbl>
## 1 rmse    standard   3316.
## 2 rsq     standard      0.000975
## 3 mae     standard   1496.
```

3.2.3 Now make one model for each cryptocurrency.

Code adapted from: https://r4ds.had.co.nz/many-models.html

First I group the data

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

```
crypto_data_grouped
```

```
## # A tibble: 2 x 2
## # Groups:   symbol [2]
##   symbol data
##   <chr>  <list>
## 1 BTC    <tibble [3,502 x 40]>
## 2 ETH    <tibble [1,625 x 40]>
```

Make a helper function with the model:

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

Now we can use `purrr(?)` 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:

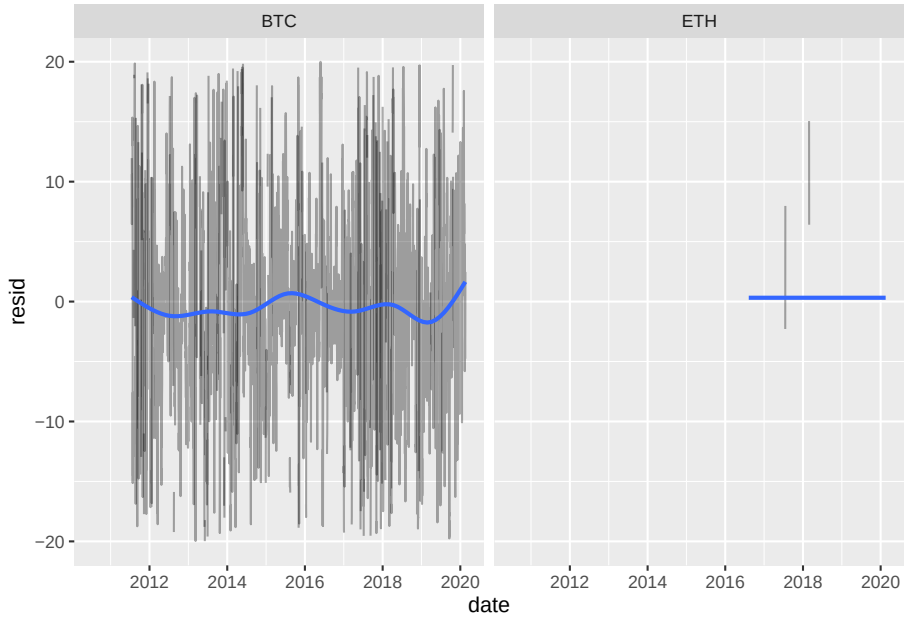
```
crypto_data_grouped
```

```
## # A tibble: 2 x 4
## # Groups:   symbol [2]
##   symbol data                  model  resids
##   <chr>  <list>                <list> <list>
## 1 BTC    <tibble [3,502 x 40]> <lm>   <tibble [3,502 x 41]>
## 2 ETH    <tibble [1,625 x 40]> <lm>   <tibble [1,625 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)
```



3.2.4 Add error metrics using `broom` (?):

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

Sort by the best scores:

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

```
## # A tibble: 2 x 15
## # Groups:   symbol [2]
##   symbol data  model resids r.squared adj.r.squared  sigma statistic  p.value
##   <chr>  <lis> <lis> <list>     <dbl>         <dbl>  <dbl>     <dbl>    <dbl>
## 1 BTC    <tib~ <lm>  <tibb~    0.158        0.148     12.4     16.1  4.47e-90
## 2 ETH    <tib~ <lm>  <tibb~    0.0341       0.00552 3203.       1.19 1.99e- 1
## # ... with 6 more variables: df <int>, logLik <dbl>, AIC <dbl>, BIC <dbl>,
## #   deviance <dbl>, df.residual <int>
```

3.2.5 How much better do the models get if we add more variables?

Add MA, EMA, etc...

3.2.6 Next Steps:

I won't go further than this here, but as my next steps, here is what I would do:

1. Use parsnip + purrr to iterate through lots of predictive models
2. How much better do the models get with hyperparameter tuning?
3. Visualize the best model before and after parameter tuning and then do the same with the worst performing model

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 will keep going with the code from the tidymodels project example I just finished and use Python for … ADD GOAL HERE …

DEV:

Replace this with the Python one:

Could make a very simple xgboost model maybe?

Could also show using Shrimpy API to pull latest data, manipulate in pandas and visualize

Mention experience/courses taken in Python and how it's never clicked with me very much but how I am taking a basic Python course in my Master's in Data Science and I am looking to take it as an opportunity to create a lot of content using reticulate.

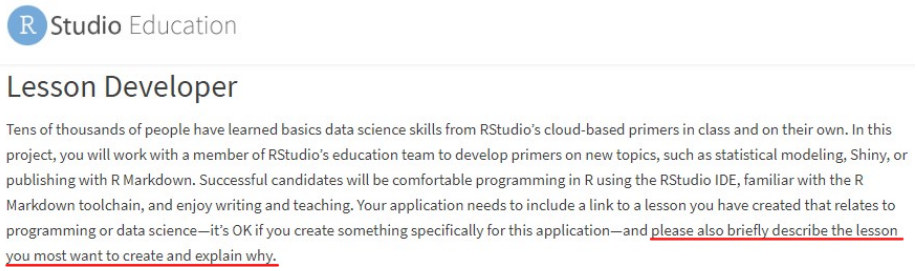
```
library(reticulate)
```

Chapter 4

About Me

Chapter 5

Ideal Tutorial



Chapter 6

Cool Charts

6.1 Disable while working on bookdown, takes too long to render!

Here are some examples of charts, which refresh daily using GitHub actions and Netlify for automation.

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