TECHACADEMY

Data Science Project Guide: Honey and Bees

TechAcademy e.V.

Summer Term 2022

Contents

1 Welcome!

In the first few chapters you will be introduced to the basics of the R and Python tracks respectively and you will find helpful explanations to questions you might have in the beginning of your coding journey. There will be a quick introduction to the Data Science track for you to get started with the project quickly. Let's start with the basics! In all tracks you will work on your project in small groups with your peers. Not only will this help you to get the project done faster, but we also want you to compare your results and discuss your findings – remember, coding is just half the effort, understanding what the data and the output shows is even more important! Our experience shows: The different backgrounds of the members and discussing different opinions and ideas will produce the best results. Besides, it is way more fun to work on a project together than to code alone!

The groups consist of four team members. You can choose your teammates independently; we will not interfere with your arrangements. It is important that all group members complete the same level of difficulty (beginner or advanced), since the tasks are different in scope for each level. We explicitly encourage you to collaborate with students from different university faculties. This not only allows you to get to know people from other faculties, but may also give you a whole new perspective on the project and tasks. When submitting your project please note: for a certificate, each person must submit the project individually. However, hand-ins may be similar within your group. You can get more information at our first Coding Meetup on **May 18, 2022.**

Every semester, the Product Development Team at TechAcademy thinks hard about a new project that is suited for you to learn data science. This semester we decided to analyse a simple yet important part of our lives: honey and bees. While honey might not be the reason why we still exist on this planet, bees very well are, hence why honey production may also be. However, with climate change and increasing globalisation, bee colonies are threatened daily. Though none of us are experts on bees and climate, we still believe that together we can all learn a lot from this case study.

This case study and the associated project guide was developed and written entirely from scratch by TechAcademy's Data Science team. Inga Lasys and Isabel Schnorr developed the project in `R`, while Thilo Leitzbach developed it in `Python`.

2 What's Data Science and How Do I Do It?

2.1 What's R?

R is a programming language developed by statisticians in the early 90s to calculate and visualize statistical results. A lot has happened since then, and by now, R is one of the most widely used programming languages in Data Science. You don't have to compile your 'R' code, but you can use it interactively and dynamically. Such an approach makes it possible to quickly gain basic knowledge about existing data and display it graphically.

R offers much more than just programming. The language provides a complete ecosystem for solving statistical problems. A large number of packages and interfaces are available, which you can use to expand the basic functionality of the programing language to, say, create a COVID-Tracker application.

2.1.1 RStudio Cloud

Before you can use R, you usually have to install some separate programs locally on your computer. Typically, you first install a "raw" version of R. In theory, you can then start programming. However, it is challenging to carry out an entire project with a "raw" version of R. That's why there is RStudio, a free Integrated Development Environment (IDE) for R.

Such IDE includes many essential features that simplify programming with R. Among other things, an auto-completion of your code, a friendly user interface, and many expansion options. Think of R as your car's engine. And think of RStudio as your car's dashboard that shows fancy metrics, has a radio and allows you to adjust air-conditioning!

Experience has shown that installing R and RStudio locally on your computer takes some effort. Fortunately, RStudio also has a cloud solution that eliminates these steps: RStudio Cloud. You can edit your project in the same IDE in the browser without any prior installations on your computer. You can also easily switch your project from a private to a public project and give your team an insight into your code via a link or by giving them access to the workspace directly. In this way, you can easily exchange ideas with your team.

We will introduce RStudio Cloud and unlock access to our workspace on our first Coding Meetup. Until then, focus on learning the "hard skills" of programming with the courses on DataCamp. That brings us to your curriculum in the next section!

2.1.2 Curriculum

The following list shows the required DataCamp courses for the Data Science with R Track at TechAcademy. As a beginner, please stick to the courses of the "beginner" program. Ambitious beginners can, of course, take the advanced courses afterward. However, it would be best if you worked through the courses in the order we listed them.

The same applies to the advanced courses. Here, too, you should finish the specified courses in the given order. Since it can, of course, happen that you have already mastered the topics of an advanced course, you can replace some courses. If you are convinced that the course does not add value to you, feel free to replace it with one of the courses in the "Exchange Pool" (see list below). However, you should not pursue an exchange course until you finish all chapters from the advanced course: "Intermediate R."

To receive the certificate, both beginners and advanced learners must complete at least two-thirds of the curriculum (6/9 courses). For the beginners, this means until – and including – the course "Data Visualization with ggplot2 (Part 1)" and for the advanced until –and including – "Supervised Learning in R: Classification." In addition, you should complete at least two-thirds of the project tasks. After completing the curriculum and the project's (minimal) requirements, you will receive your TechAcademy certificate!

R Fundamentals (Beginner Track)

1. Introduction to R (4h)
2. Intermediate R (6h)
3. Introduction to Importing Data in R (3h)
4. Cleaning Data in R (4h)
5. Data Manipulation with dplyr (4h)
6. Data Visualization with ggplot2 (Part1) (5h)
7. Exploratory Data Analysis in R (4h)
8. Correlation and Regression in R (4h)
9. Multiple and Logistic Regression in R (4h)

Machine Learning Fundamentals in R (Advanced Track)

1. Intermediate R (6h)
2. Introduction to Importing Data in R (3h)
3. Cleaning Data in R (4h)
4. Importing & Cleaning Data in R: Case Studies (4h)
5. Data Visualization with ggplot2 (Part1) (5h)
6. Supervised Learning in R: Classification (4h)
7. Supervised learning in R: Regression (4h)
8. Unsupervised Learning in R (4h)
9. Machine Learning with caret in R (4h)

Data Science R (Advanced Track) – Exchange Pool

- Data Visualization with ggplot2 (Part 2) (5h)
- Interactive Maps with leaflet in R (4h)
- Machine Learning in Tidyverse (5h)
- Writing Efficient R Code (4h)
- Support Vector Machines in R (4h)
- Supervised Learning in R: Case Studies (4h)
- Optimizing R Code with Rcpp (4h)

2.1.3 Helpful Links

- RStudio Cheat Sheets

- RMarkdown Explanation (to document your analyses)
- StackOverflow (forum for all kinds of coding questions)
- CrossValidated (Statistics and Data Science forum)

2.2 What's Python?

`Python` is a dynamic programming language. You can execute the code in the interpreter, so you do not have to compile the code first. This feature makes `Python` very easy and quick to use. The excellent usability, easy readability, and simple structuring were and still are core ideas in developing this programming language.

You can use `Python` to program according to any paradigm, whereby structured and object-oriented programming is most straightforward due to the structure of the language. Still, functional or aspect-oriented programming is also possible. These options give users significant freedom to design projects the way they want and great space to write code that is difficult to understand and confusing. For this reason, programmers developed specific standards based on the so-called `Python` Enhancement Proposals (PEP) over the decades.

2.2.1 Anaconda and Jupyter

Before you can use `Python`, you must install it on the computer. `Python` is already installed on Linux and Unix systems (such as macOS), but often it is an older version. Since there are differences in the handling of `Python` version 2 – which is no longer supported – and version 3, we decided to work with version 3.6 or higher.

One of the easiest ways to get Python and most of the best-known programming libraries is to install Anaconda. There are detailed explanations for installing all operating systems on the website of the provider.

With Anaconda installed, all you have to do is open the Anaconda Navigator, and you're ready to go. There are two ways to get started: Spyder or Jupyter. Spyder is the integrated development environment (IDE) for `Python` and offers all possibilities from syntax highlighting to debugging (links to tutorials below).

The other option is to use Jupyter or Jupyter notebooks. It is an internet technology-based interface for executing commands. The significant advantage of this is that you can quickly write shortcode pieces and try them out interactively without writing an entire executable program. Now you can get started!

If you have not worked with Jupyter before, we recommend that you complete this DataCamp course first. There you will get to know many tips and tricks that will make your workflow with Jupyter much easier.

To make your work and, above all, the collaboration more accessible, we are working with the Google Colab platform that contains a Jupyter environment with the necessary libraries. You can then import all the data required for the project with Google Drive. We will introduce this

environment during our first Coding Meetup. Until then, focus on learning the "hard skills" of programming with your courses on DataCamp. This topic brings us to your curriculum in the next section!

2.2.2 Curriculum

The following list shows the required DataCamp courses for the Data Science with `Python` Track at TechAcademy. As a beginner, please stick to the courses of the "beginner" program. Ambitious beginners can, of course, take the advanced courses afterward. However, it would be best if you worked through the courses in the order we listed them.

The same applies to the advanced courses. Here, too, you should finish the specified courses in the given order. Since it can, of course, happen that you have already mastered the topics of an advanced course, you can replace some courses. If you are convinced that the course does not add value to you, feel free to replace it with one of the courses in the "Exchange Pool" (see list below). However, you should not pursue an exchange course until you finish all chapters from the advanced course: "Intermediate Python."

To receive the certificate, both beginners and advanced learners must complete at least two-thirds of the curriculum (6/9 courses). For the beginners, this means until – and including – the course "Joining Data with pandas (4h)" and for the advanced until –and including – "Exploratory Data Analysis in Phyton (4h)." In addition, you should complete at least two-thirds of the project tasks. After completing the curriculum and the project's (minimal) requirements, you will receive your TechAcademy certificate!

Python Fundamentals (Beginner Track)

1. Introduction to Data Science in Python (4h)
2. Intermediate Python (4h)
3. Python for Data Science Toolbox (Part 1) (3h)
4. Introduction to Data Visualization with Matplotlib (4h)
5. Data Manipulation with pandas (4h)
6. Joining Data with pandas (4h)
7. Exploratory Data Analysis in Phyton (4h)
8. Introduction to DataCamp Projects (2h)
9. Introduction to Linear Modeling in Python (4h)

Data Science with Python (Advanced Track)

1. Intermediate Python (4h)
2. Python Data Science Toolbox (Part 1) (3h)
3. Python Data Science Toolbox (Part 2) (4h)
4. Cleaning Data in Python (4h)
5. Exploring the Bitcoin Cryptocurrency Market (3h)
6. Exploratory Data Analysis in Phyton (4h)
7. Introduction to Linear Modeling in Python (4h)
8. Supervised Learning with Scikit-Learn (4h)
9. Linear Classifiers in Python (4h)

Data Science with Python (Advanced Track) - Exchange Pool

- TV, Halftime Shows and the Big Game (4h)
- Interactive Data Visualization with Bokeh (4h)
- Time Series Analysis (4h)
- Machine Learning for Time Series Data in Python (4h)
- Advanced Deep Learning with Keras (4h)
- Data Visualization with Seaborn (4h)
- Web Scraping in Python (4h)
- Writing Efficient Python Code (4h)
- Unsupervised Learning in Python (4h)
- Writing Efficient Code with pandas (4h)
- Introduction to Deep Learning in Python (4h)
- ARIMA Models in Python (4h)

2.2.3 Helpful Links

Official Tutorials/Documentation:

- https://docs.python.org/3/tutorial/index.html
- https://jupyter.org/documentation

Further Explanations:

- https://pythonprogramming.net/
- https://automatetheboringstuff.com/

- https://www.reddit.com/r/learnpython
- https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook

2.3 Your Data Science Project

Participant's Section:

- https://www.tech-academy.io/teilnehmerbereich

2.3.1 Prelude

- Kick-Off Workshop (27.04.2022)
- `R` / `Python` Coding Introduction (04.05.2022)

2.3.2 Coding Meetups

- **18.05.2022**, Coding MeetUp 1
- **01.06.2022**, Coding MeetUp 2
- **15.06.2022**, Coding MeetUp 3
- **29.06.2022**, Coding MeetUp 4

3 Introduction to Your Project

3.1 Purpose of the Project Guide

Welcome to the project guide for your TechAcademy data science project! This document will guide you through the different steps of your project and will provide you with useful hints along the way. However, it is not a detailed step by step manual, since we feel like it is important that you develop the skills of coming up with your own way of solving different tasks. This is a great way to apply the knowledge and tools you have acquired in Data Camp, as well as developing skills in researching code independently. Always remember, Google is the best tool in your kit! It might happen that you don't know how to solve a task. This is a normal part of the coding process, so don't worry. It is part of the learning experience and we provided you with helpful tips throughout this guide. We also included pictures of what your results could look like. They are meant to be a useful guidance so that you know what you are working towards. Your plots do not have to look the same way as ours do, just keep in mind which information is necessary for a meaningful plot. Furthermore, you can find helpful links in the introductory chapters, where your questions might already have been answered. If not, and in the unlikely case that even Google cannot help you, the TechAcademy mentors will help you via Slack or directly during the coding meetups. At the end of the project guide you will find an overview of all tasks that have to be completed, depending on your track (beginner/advanced). You can use this list to check which tasks still need to be completed and which tasks are relevant for your track.

3.2 What is this Project About?

3.3 Exploratory Data Analysis – Getting to Know the Data Set

3.4 Prediction – Apply Statistical Methods

4 Exploratory Data Analysis

Before you can dive into the data, set up your programming environment. This will be the place where the magic happens - all your coding takes place there.

In your workspace on RStudio Cloud, we have already uploaded an "assignment" for you (Template HoneyAndBees). When you create a new project within the workspace *Class of '22 | TechAcademy | Data Science with R*, your workspace will open up.
We've already made some arrangements for you: The data sets you will be working with throughout the project are already available in your working directory. We also created an RMarkdown file (a file that ends with `.Rmd` extension), with which you will be able to create a detailed report of your project. You can convert that file into an HTML document when you have finished coding the project in R. Open the file `Markdown_HoneyAndBees.Rmd` and see for yourself!

We recommend using Google Colab for this project since it requires no particular setup, stores its notebooks to your Google Drive, and makes it easy for you to share them with your team members.
As an alternative to Google Colab, you might want to install Jupyter Notebook locally using the Anaconda distribution. We will give you a more detailed step-by-step demo during the first coding meetup.

Next up is importing the data sets. It's best to this **once** on the top level of your notebook / script. You can always copy to new variables and work on slices of the data frames afterwards.

- Honey
- Bees
- Weather

4.1 Exploring Honey and Bees Data

4.1.1 Discovering Formats, Frequencies, and Fuck-ups

Let's start by looking at the bee & honey data sets.

- Have a look at the frequency of the records (is there a difference across sets?)
- Do you see any missing values or data entries that are different from the other entries?
- What are the data formats (e.g. data types like strings or floats but also date formats such as yearly)?
- Look at the unit of every variable, can you make sense of the units?

Write down a couple of sentences to these questions. Also, comment on any errors / difficulties which you notice and that could be an issue later on in the project. No worries, there is no right or wrong.

You can import the data directly from your working directory. Use `head()`, `str()`, or `class()` to get an overview.

You can feed the links to the respective data files above to a method of the pandas package (you might want to specify the index column). Check out the resulting `pd.DataFrame` instance with the `head()` method and the `dtpyes` attribute. You can also dig into a specific column with `describe()`. Here's an additional pandas cheat sheet for you to reference

4.1.2 Give some overall statements

Lets stick with the honey data. Since you have already got a feeling by now, it would be interesting to indicate some outstanding features of the set:

- Which state had the most producing colonies ever and in which year? Anything special about that state?
- What state had the lowest price for honey. How low was it?
- What is the total Honey production for 2016?

4.2 Data Cleaning and Useful Transformations

The exercises in this and the next section are about the bees data set. On a later stage of the project we would like to merge it with the honey data set. To this end, we will now spend some time preparing the bees data.

4.2.1 Date Formatting

Remember, one of the differences is the frequency of the data. The date (format) of the bees set is quarterly whereas we have yearly honey data. Recall also that dates, even just years, can be formatted differently than just as a string or integer. Maybe you have checked it before, but now is definitely the moment to translate the `Date` column to a fitting date format.

Add tip

`pd.to_datetime()`

4.2.2 More Data Types - Strings, Ints, and Floats

So far you converted years into a more appropriate date format for further investigation. Now, look at the other columns of the bees data set: Some appear to contain numbers but are they also of a numeric data type? Check out all columns so you don't miss out on one!

For further calculations, it is safer to convert these columns to such a numeric data type. While there are vectorized methods that can convert data types at once, we would like you to write a simple (really) 'for' loop to conduct the conversions.

Add tip

You need an iterable to loop over, the semicolon after the keyword, and the indentation of the subsequent lines! Here is a cheat sheet in case you have forgotten.

4.2.3 Converting Units

From the column names you can see that the honey data set is in pounds as we are dealing with an U.S. data set. However, we are devoted followers of the metric system and do not understand pounds. Consequently, we would like to translate them to kilogram [kg]. While you are at it, you might want to give your new columns in kg a shorter name with less spaces in it …

4.2.4 Rounding

Have a look at your column values. Some columns have long decimal numbers. To simplify your data frame, round up to the decimal place you prefer. Please write one or two sentences why you think rounding is O.K. and why you chose the decimal place X or even a conversion to ints.

4.3 Simple Metrics

4.3.1 Missing Values

Look at the bees set and its missing values. Decide on how you want to treat them and give us a detailed explanation why you have decided to treat the missing values the way you did. Keep in mind that these transformations affect all your subsequent work on the data and may bias your findings. Sometimes you are lucky and can also identify a pattern that will ease your pain of treating the values and deciding for a strategy.

There are many possibilities: From excluding the rows to forward / backward fills or even inserting an average / median.

We highly recommend the Datacamp course on missing values in R for this exercise.

We highly recommend the Datacamp course on missing values in Python for this exercise.

4.3.2 Lost Colonies

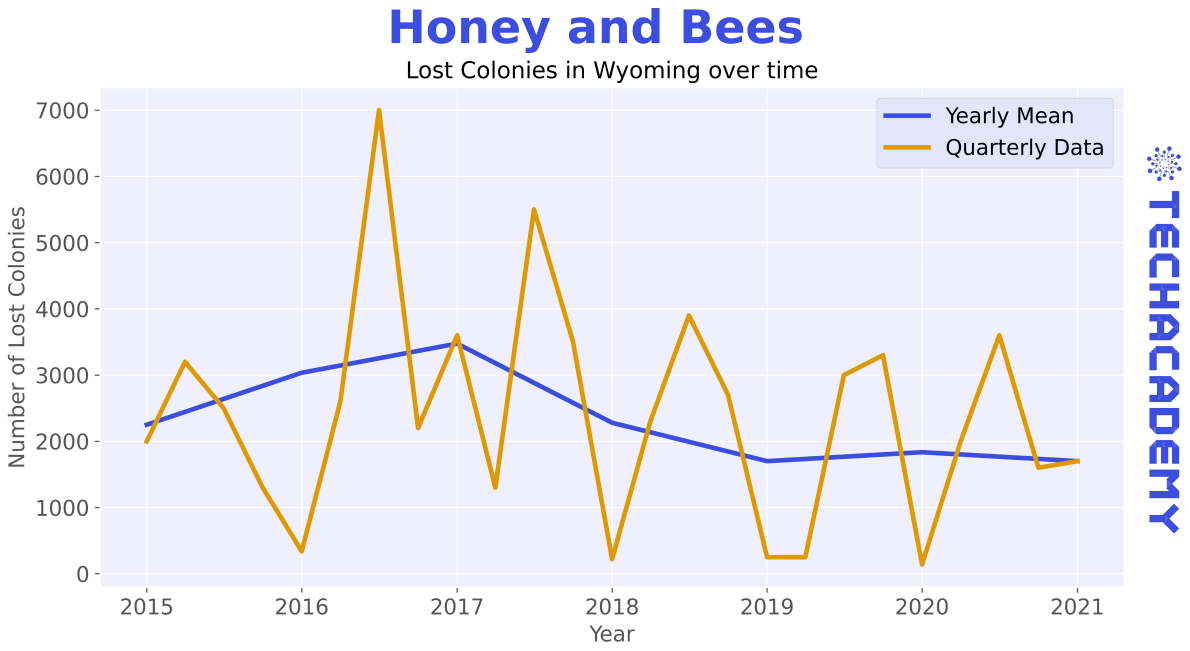
Calculate the percentage of lost colonies (it is as simple as it reads).

4.3.3 Group Means

We want to merge the honey and bees data set to analyze them jointly. To do so we need to reduce the frequency of the bees data frame (quarterly) to yearly. Accordingly, you would like to group by state and quarter and replace the observations by an aggregate. You decide to replace with the mean (over the four quarters). Please mind how you are rounding, print the table of your aggregated data frame, and answer the following two questions

- What are potential hazards of reducing data like this?
- What other measures could you use?

You may refer to the following graph.



There is no absolute answer to these questions since your approach most likely depends on what you are analyzing / trying to show or understand.

Add tip

Add tip

4.4 Merging Data Sets

When preparing to data sets, it is wise to check the naming of columns again. Please give your "bees" data columns appropriate names (we recommend names where empty spaces are replaced with an underscore "_"), if you have not done so for the honey data set please do the same.

Add tip

Column names can be replaced very neatly by passing a dictionary `{"x": "y", }` to a certain pandas method.

4.4.1 Joint Honey and Bees Set

Do an inner merge of the two sets by exploiting the index / State names and the respective date columns. Closely compare the sizes of the individual data frames and of the aggregate one.

- Do they match (*hint: they do not*)?
- Why not and what states are missing from which individual data frame?

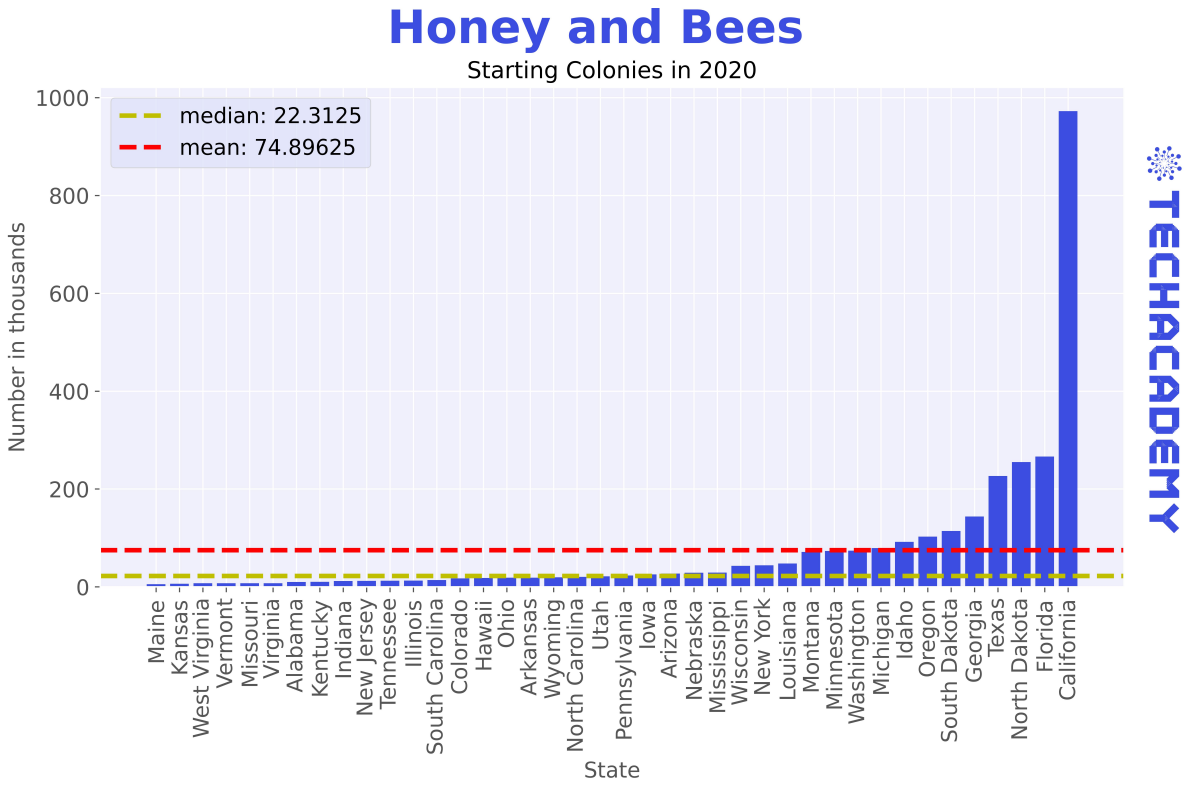
Add tip

Add tip

4.5 States Bar Plot

Use the aggregate (yearly) set and create a bar plot with all states plotting starting bee colonies from the lowest to the highest value. Please also include the average and the median of the observations.

You can decide for yourself whether you want to aggregate over all years or over a single year of your choice.



Add tip

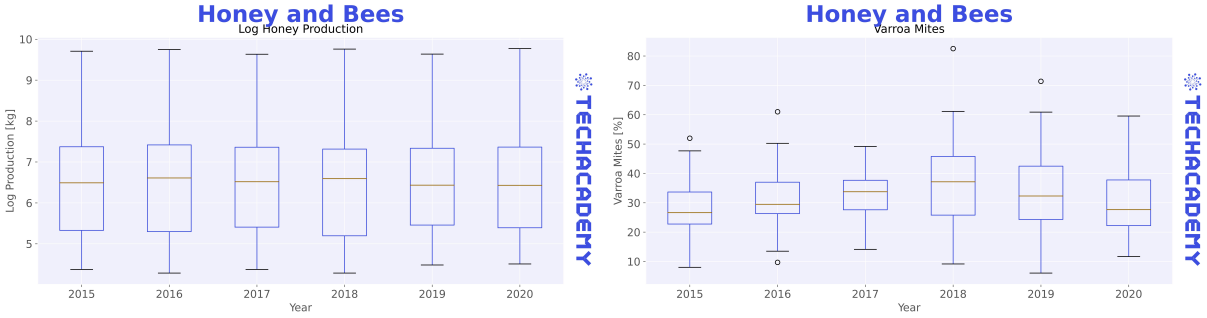
Add tip

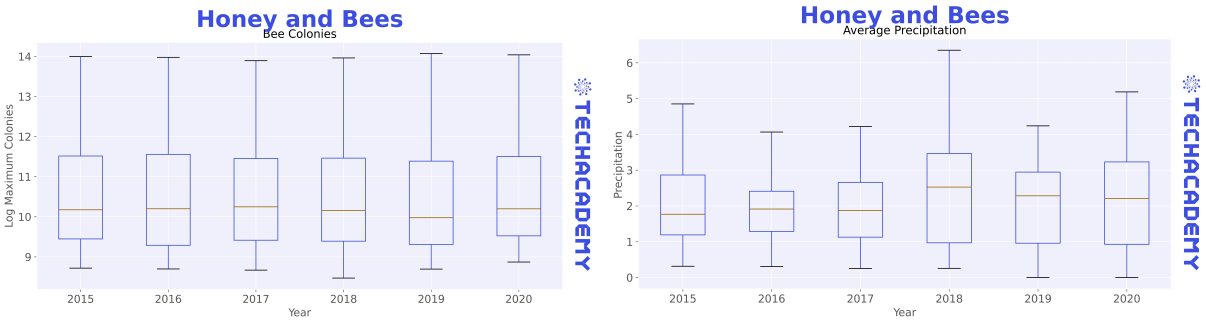
4.5.1 Splitting the Data Set

Split a subset for data between the X different Team members (we have 40 States) (ungraded)

4.6 Yearly Boxplots

Boxplot aggregated for year: Honey Production, Parasite: Varroa Mite, Precipitation , Bee Colonies (6 boxen).



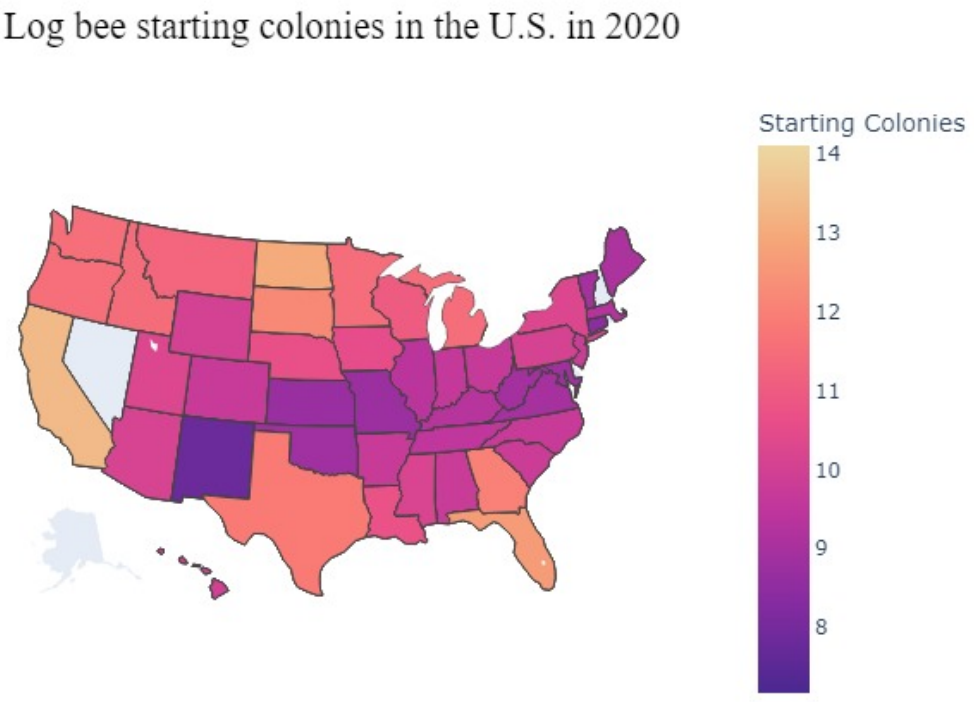


Add tip

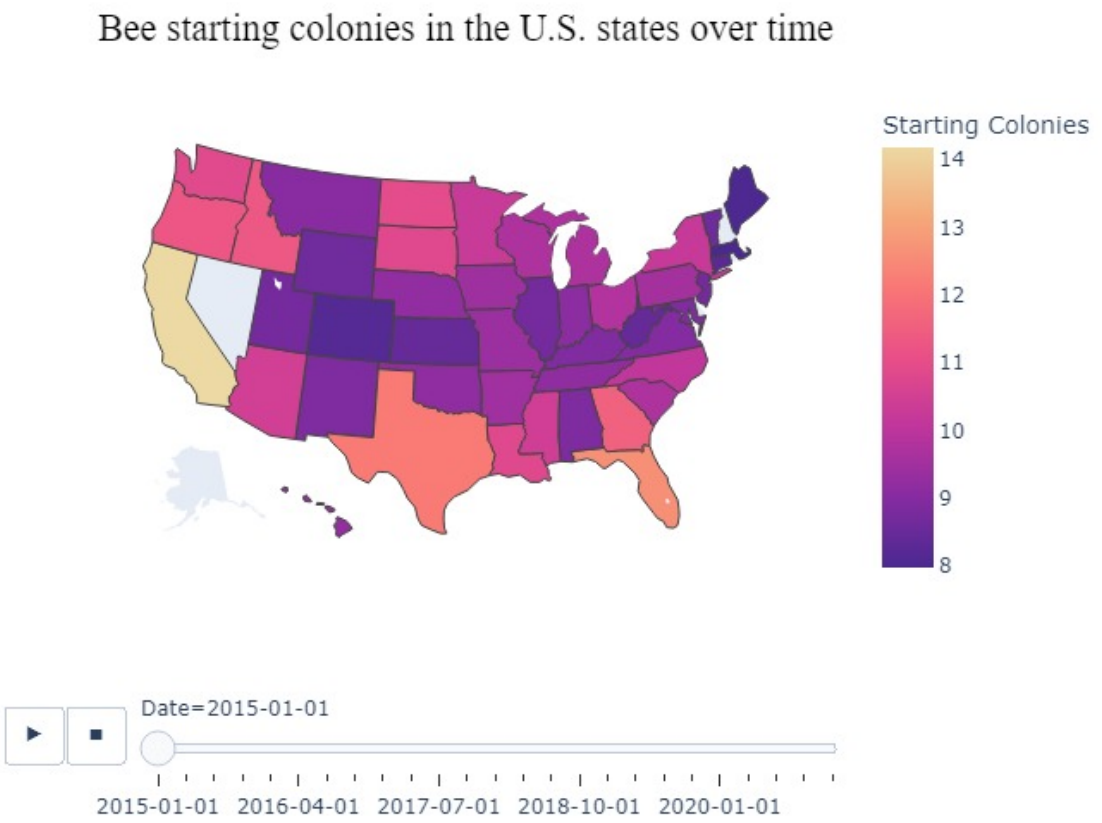
Add tip

4.7 Map with State Popups

Last but not least we want to do a more appealing visualization. The idea is to combine the geographic information on the states with the numeric data to provide a rich visual that reflects the two combinations.



As a bonus, you can also animate the transitions over time for each state creating a very dynamic plot.



Add tip

Add tip

Congratulations! Based on your work with fundamental data transformations and many visualizations, you now have a solid understanding of the honey and bees data sets! With this, you have completed the *EDA* part of the project!

5 Seasonality and Geographic Differences – Application of Statistical Methods

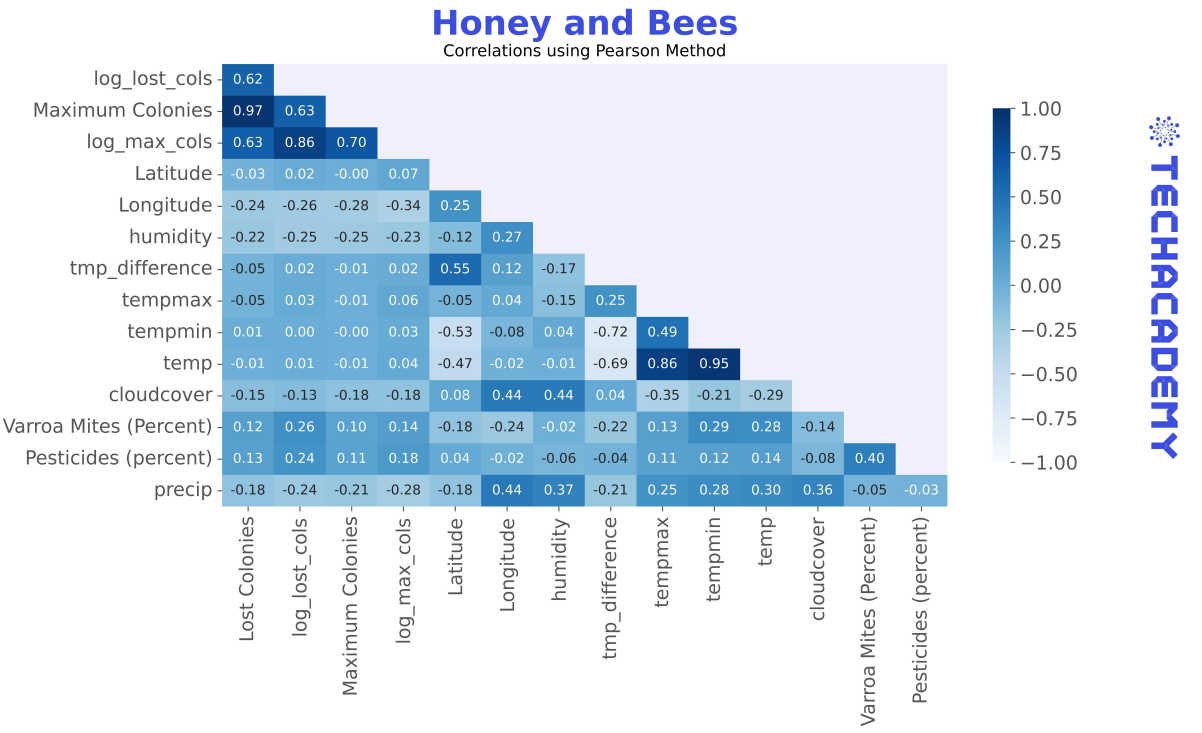
5.1 Merging Bees and Weather

We want to investigate how diseases and different climates affect our beloved bee populations. To that end, merge the two data sets. Be careful to aggregate the weather data set wisely to quarterly data. It might not be clever to replace period-specific max / min values with a mean.

5.2 Correlations

5.2.1 Naive Approach

Do a simple correlations plot between variables of interest like lost colonies and different parasites and weather indicators. You could also create your own indication such as a temperature differential or interaction effects.



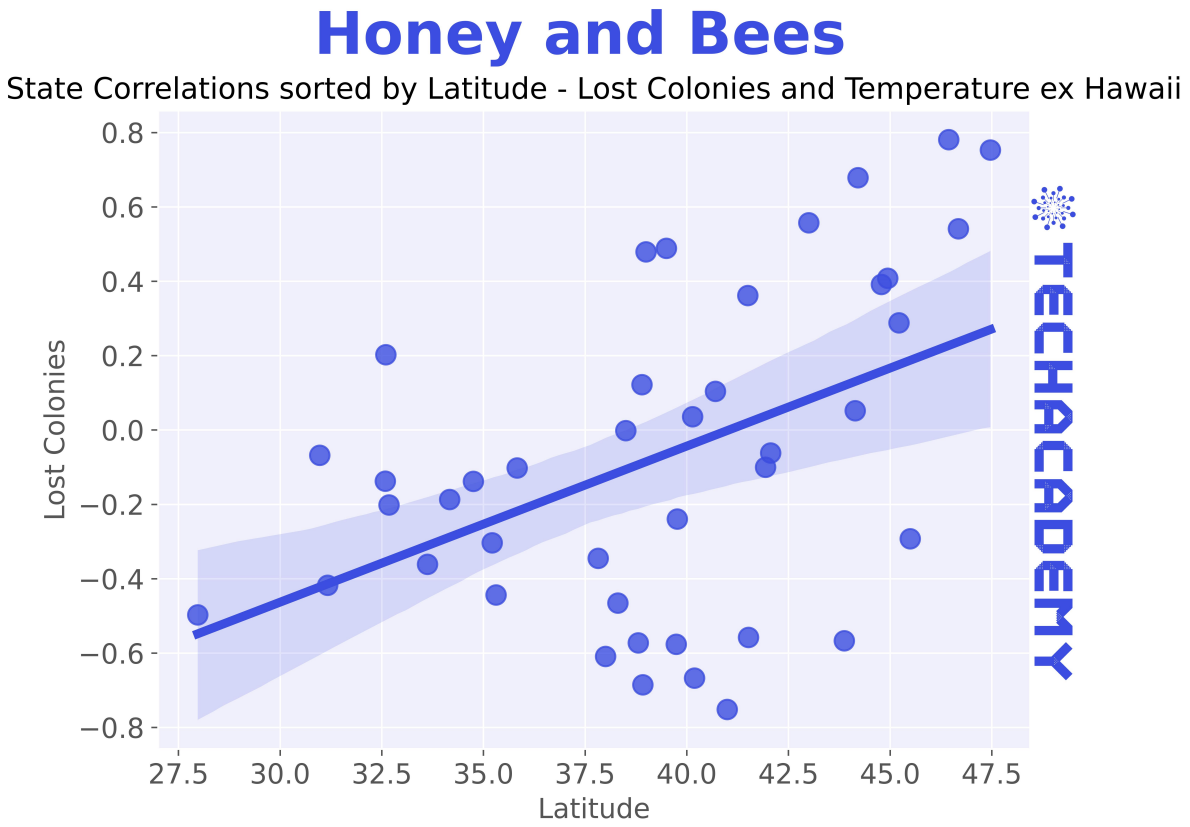
- Why is a naive correlation matrix across states a terrible idea?

Think for example about omitted variables and how information is lost when you aggregate fundamentally different groups.

Add tip

Add tip

5.2.2 Statewise Correlations



Add tip

Add tip

5.3 Simple Linear Models for State Panel Data

5.4 Exploiting Geographic Differences

5.5 Seasonality

6 Exercise Checklist

This checklist should help you keep track of your exercises. Remember that you have to hand in satisfactory solutions to at least two-thirds of the exercises. If you're part of the beginner track, this refers to two-thirds of part A (EDA) only. If you're part of the advanced track, you have to hand in at least two-thirds of both individual parts A and B. Hence, you cannot hand in 100 percent of the first part and only 50 percent of the second one. You'll need more than 66% in each one for a certificate. After all, you're not that advanced if you only did half of it, right?

Part 1: Exploratory Data Analysis (Beginner + Advanced Tracks)

Part 2: Prediction Using Statistical Methods (motivated Beginner + Advanced Tracks)

7 What's Next in Your Data Science Career?

7.1 Data Science in General

Version Control with Git

Advice for Non-Traditional Data Scientists

Learn from Great Data Scientists on Kaggle

-> add towardsdatascience

7.2 R

Install R and RStudio Locally
RStudio.Cloud is great for getting started with R without having to worry about installing anything locally. Sooner or later you will have to install everything on your own computer. Here's a DataCamp tutorial on how to do that.
Version Control with Git
RStudio has a nice interface that lets you enjoy the perks of Git without ever having to touch the command line – sounds great, does it? Learn how to set up the Git & R workflow with Happy Git with R.
R Graph Gallery
Get inspiration to take your plotting to the next level. Includes code to reproduce the plots.
Follow the R Master Himself and the R Community
Hadley Wickham was and continues to be extremely influential on the development of R and its rise to one of the most popular data science languages. He's behind many tools that we taught you in this semester, especially the tidyverse (including great packages such as ggplot2 and dplyr). Follow him on Twitter to get great R advice and keep up to speed with everything new to R. Following the many people behind R (not only Hadley) is a great way for acquiring deeper understanding of the language and its developments.
Join the Campus useR Group in Frankfurt
There's a quite active R community in Frankfurt that meets once a month. It's open for students, professors, industry practitioners, journalists, and all people that love to use R. In those meetings, you'll hear about other's work, discuss new developments, and ask questions.
Listen to R Podcasts Another great way to easily keep up with new developments in the Data Science/R community. Check out Not So Standard Deviations or the R-Podcast

7.3 Python

Install Python Locally

Until now you've only programmed using JupyterHub on the TechAcademy Server. A next step would be to install Python and Jupyter locally on your computer. This link contains the necessary information on how to install the software on Windows, iOS or Linux.

Choosing the Right Editor

Using Jupyter is especially useful for short data analyses. But sometimes you want to write longer scripts in Python. In these cases, it is often more convenient to use a code editor instead of Jupyter. This tutorial highlights the positive aspects of such an editor and how to choose the right one for you. Pro-Tip: Also check out the other tutorials on Real Python and check out the Community Version of "PyCharm" which is the most common Python IDE (Integrated Development Environment.

Python Graph Gallery

Get inspiration to take your plotting to the next level. Includes code to reproduce the plots.

More Advanced Python Concepts

You know the basic data structures in Python like lists and dictionaries. What are the next steps to improve your knowledge? This website gives good explanations for slightly more advanced concepts which can be very useful from time to time.

A Deeper Understanding

If you want to get a deeper understanding of the Python programming language and into typical algorithms which are used in the field of Data Science, this free book can be a good starting point.

Writing Beautiful Python Code

"My code doesn't look nice, but it works!" This might work for yourself, but often you will work on code with other people. But even if you're just coding for yourself it's a good idea to follow the PEP8 style guide. It's a useful convention on how to structure and code in Python. You'll find useful resources for PEP8 here and here.

Listen to Python Podcasts

When you don't have time for books you can listen to Talk Python or the Python Podcast.