

# STAT 471: Final Project Instructions

Eugene Katsevich

November 8, 2021

## Contents

<b>1</b>	<b>Overview</b>	<b>1</b>
1.1	Format . . . . .	1
1.2	Logistics . . . . .	1
1.3	Deliverables . . . . .	2
<b>2</b>	<b>Guidelines</b>	<b>2</b>
2.1	Data . . . . .	2
2.2	Exploratory data analysis . . . . .	2
2.3	Data mining . . . . .	2
2.4	Documentation . . . . .	2
<b>3</b>	<b>Recommended workflow</b>	<b>3</b>
3.1	Step-by-step workflow . . . . .	3
3.2	Template . . . . .	3
3.3	Getting started . . . . .	3
3.4	Working on your project . . . . .	5
<b>4</b>	<b>Final report</b>	<b>5</b>
<b>5</b>	<b>Pitfalls</b>	<b>6</b>

## 1 Overview

The goal of the final project is to apply the tools you’ve learned in class to a real-world data mining problem.

### 1.1 Format

You will find and/or assemble a dataset relevant to a domain that interests you (finance, business, sports, healthcare, etc.), and formulate one or more analysis goals that can be addressed based on the data. You will carry out the data mining tasks we learned in class: data import, cleaning, and exploration followed by building, interpreting, and evaluating predictive models. After the data mining is done, you will make a set of conclusions and recommendations based on the analysis.

### 1.2 Logistics

The final project can be completed individually or in groups of two. Only one submission per group is necessary. Final project reports will be submitted through Gradescope, with a link to the underlying Github repository on the front page. All deliverables are due on Sunday, December 19 at 11:59pm. Final projects will be accepted up to three days late, with a 5 point deduction per late day. Students with unused late passes for homework can use these on the final project instead; late penalties will be assessed on a per-student basis

rather than a per-group basis. The teaching staff are happy to provide guidance on any aspect of your final projects, so feel free to reach out to us through email, Piazza, or office hours.

### 1.3 Deliverables

The primary deliverable will be a final report presenting the problem, data, analysis, and conclusions in a way that would be clear to a technically literate stakeholder (e.g. your manager at a tech company). Unlike the homeworks, this report will not contain any code; its main goal is to communicate your analysis and your results. It will be evaluated based on the high standard of presentation quality outlined in [preparing-reports.pdf](#). In addition to the final report, you will submit the underlying data and code in the form of a Github repository. More details on these deliverables are provided below.

## 2 Guidelines

### 2.1 Data

Your final dataset must contain at least 500 observations and at least 15 features (though it may contain way more of either). You may need to merge data from multiple sources to address the analysis goal(s) you propose. The response variable can be continuous or discrete. Your data set may not already have undergone cleaning for data mining purposes; for example, you may not draw data from [Kaggle](#), the [UCI Machine Learning Repository](#), or other sources of clean datasets curated for data science / machine learning. As a starting point, you may want to browse the [Awesome Public Datasets repository](#).

### 2.2 Exploratory data analysis

You must conduct a thorough exploratory data analysis on your data, including visualizations and summary statistics. It is recommended that you conduct your exploratory data analysis on the training data only, but you may also explore the entire dataset to an extent before splitting it. For example, in a classification problem the class proportions might impact your choice of how to split the data. Just make sure that any exploration you do on the entire dataset (as opposed to just the training set) does not impact your choice of statistical machine learning procedure. For example, you cannot select highly predictive features based on data that includes the test data. It is ok, however, to inspect the response distribution or relationships among features based on the entire data.

### 2.3 Data mining

You must apply at least three distinct statistical machine learning methodologies to the data across at least two of the major classes of methods learned in STAT 471 (regression-based methods, tree-based methods, and deep learning). Each analysis should be accompanied by detailed results and interpretation in the context of the application domain.

### 2.4 Documentation

It is crucial that data analyses be reproducible, so that other data scientists can understand their details as well as replicate and extend them. To this end, you will submit all your code and data in the form of a Github repository. This repository should be well-organized (a template has been provided for you; see [Section 3.2](#)), and a file named `run-all.R` in this repository should reproduce all your results from scratch, starting from dataset download. (Ideally, you should have download links for each raw dataset, and R scripts that download the raw data based on these links. This may not always be possible, in which case you should manually download the raw data and describe in as much detail as possible how you did so.)

## 3 Recommended workflow

### 3.1 Step-by-step workflow

It is recommended that you follow the step-by-step workflow (Figure 1). In this workflow, analysis tasks are modularized in their own R scripts or R Markdown files. The analysis scripts rely on each other's outputs, and the reporting is done in a separate R Markdown or regular word processor file. Analysis results can be stored in the form of R objects using the Rda format (e.g. a fit object from a random forest), tables using tsv/csv format (e.g. a table of feature importance scores), or images using e.g. png format (e.g. a CV plot).

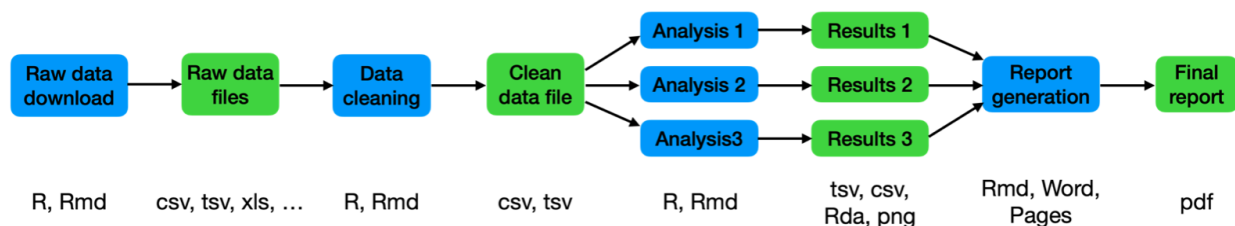


Figure 1: Step-by-step workflow. Blue boxes indicate scripts and green boxes indicate output files.

### 3.2 Template

To keep track of all the files involved in a step-by-step workflow, it is recommended to set up a folder structure with separate folders for code, data, results, and your final report. It is recommended to keep raw data and processed data in separate subfolders; the raw data should be downloaded and then never modified. These structures are illustrated in a [template repository](#) that is available to you on Github. This template repository was created by the instructor based on the [sample final project](#) for illustration purposes, but some of the analyses are omitted from the template. Please refer to the latter link for the complete report. The template repository is structured as follows:

- **data.** Contains subfolders for **raw** and **clean** data.
- **code.** Contains all analysis scripts, numbered in the order they are intended to be run, along with **run-all.R**, which runs each of these scripts. Analysis scripts are well-commented.
- **report.** Contains the Rmd file for the final report and the PDF output.
- **results.** Contains the outputs of scripts in the **code** directory in various formats, including **tsv**, **png**, and **Rda**.

There is also a **README.md** file in the root of the repository with the title, authors, date, and executive summary of the report.

### 3.3 Getting started

To get started with the above template, **one of the team members** should follow these steps:

1. Go to Github and create a new repository for your final project; this repository will be distinct from **stat-471-fall-2021**. Before clicking the green **Create repository** button, make sure your screen looks exactly like Figure 2, except your username will appear instead of **ekatsevi** and you should enter a descriptive name for your repository instead of **your-project-name**. In particular, do not check any of the three boxes at the bottom. Also, I recommend you keep your repository private while you work on your project (by choosing **Private** at in Figure 2); you will have the option to make it public when you are finished.
2. If there is a second team member, [add this team member to the repository as a collaborator](#).
3. Navigate in the Terminal to the directory within which you would like to place the directory for the final project. This might be the directory containing your **stat-471-fall-2021** folder.


### Repository template

Start your repository with a template repository's contents.

No template ▾

Owner \*

Repository name \*


 ekatsevi ▾

/


your-project-name ✓

Great repository names are short and memorable. Need inspiration? How about **bug-free-disco**?

**Description** (optional)

☐  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☒  **Private**

You choose who can see and commit to this repository.

**Initialize this repository with:**

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Figure 2: Creating a Github repository for your final project.

4

4. In the Terminal, type

```
git clone https://github.com/Katsevich-Teaching/final-project-template.git
```

This will create a folder called `final-project-template` containing all the files from the template.

5. Using your file explorer (e.g. Finder on Macs), change the name of your folder to the name of your project. Make sure it contains no spaces. Alternatively, you can do this from the Terminal by typing

```
mv final-project-template your-project-name
```

6. In the Terminal, navigate inside of your project folder by typing

```
cd your-project-name
```

7. Change the remote directory of the repository to the one you created in step 1 by typing at the Terminal:

```
git remote set-url origin https://github.com/your-username/your-project-name.git
```

8. Push all the files to Github by typing at the Terminal:

```
git push
```

If there is a second team member, he/she should follow these steps:

1. Navigate in the Terminal to the directory within which you would like to place the directory for the final project. This might be the directory containing your `stat-471-fall-2021` folder.

2. In the Terminal, type

```
git clone https://github.com/first-team-members-username/your-project-name.git
```

This will create a folder called `your-project-name` containing all the files from the template.

### 3.4 Working on your project

You will want to start by removing the files in the cloned template repository that are not relevant to your project, e.g. the data sets and the results (while keeping the directories themselves). Note that you can still refer to the original [template repository](#) on Github throughout your project. After you find your dataset(s), I recommend working step-by-step through the scripts in the `code` directory: `0-download.R`, `1-cleaning.R`, etc. Once you have generated your results, you can start putting the pieces together in `final-project-report.Rmd`. Whether you are working alone or in a group of two, I strongly recommend following the [Git best practices](#) throughout.

## 4 Final report

While the final report must be submitted in PDF form, it can be generated from either R Markdown or from a word processing software. The final report must contain the following headings and sub-headings:

- **Executive summary.** An approximately page-long summary, including
  - **Problem.** Brief problem description.
  - **Data.** Brief description of the data.
  - **Analysis.** Overview of the analyses conducted.
  - **Conclusions.** Main conclusions of the analysis, including takeaways for stakeholders.
- **Introduction**
  - **Background information.** What is the context for the data analysis?
  - **Analysis goals.** What exactly are the analysis goals? This includes a description of which features will be used to predict which response, and how success will be evaluated.
  - **Significance.** Why is this analysis goal important to address in the context of the application?
- **Data**

- **Data source(s).** Where were the data obtained? How were the data originally collected? What individual or organization collected them? A link to the data online should be included.
- **Data cleaning.** How were the data cleaned? Be specific.
- **Data description.** What is the number of observations in the data, and what does each observation represent? What is the number of features in the data? Provide a list of each feature present, its type (continuous vs categorical), and a short explanation of its meaning. What is the response variable? Is it continuous or categorical?
- **Data allocation.** How were the observations in the data allocated for exploration/training and testing?
- **Data exploration.** What is the variation in the response variable? What is the variation in the features? What is the covariation between features? What are some initial insights into the relationship between response and features?
- **Modeling**
  - **Model class 1 (e.g. regression methods).** Training, tuning, and interpretation.
  - **Model class 2 (e.g. tree-based methods).** Training, tuning, and interpretation.
- **Conclusions**
  - **Method comparison.** Which methods performed the best, and why might this have been the case?
  - **Takeaways.** What recommendations would you make to stakeholders based on your analysis? Express your answer in plain language accessible to a non-statistician.
  - **Limitations.** What were limitations of the data and/or the analysis, and what obstacles did these limitations present?
  - **Follow-ups.** What additional data collection or analysis would you recommend as a follow-up to your project?

In addition, the **link to your Github repository** should be provided on the front page of the final report.

## 5 Pitfalls

The following are issues students experienced last semester, which you should avoid:

- **Lack of independence among samples.** Supervised learning takes place in the context when independent pairs  $(X_i, Y_i)$  are available for  $i = 1, \dots, n$ . There are a few cases when this independence assumption can be broken:
  - *Time series data.* If  $i$  indexes day and we make observations  $(X_i, Y_i)$  each day, then typically samples on consecutive days will be strongly correlated. It is recommended that you time-average such data (like with the COVID example), so that  $i$  indexes time-averaged independent sampling units (e.g. case-fatality ratio for a given year for each county).
  - *Ranking problems.* Suppose you want to predict which NBA team will win the championship or which movie will win an Oscar, based on various features of these teams/movies. There is a lack of independence here because only one team can win the championship and only one movie (per category) will win an Oscar. It is best to stay away from such data or, at the very least, reformulate it so that you are actually working with independent samples.
- **Categorical features with very many levels.** Suppose you wish to predict how many Spotify downloads a song will get, using the artist as a feature. This is a categorical feature with tens of thousands of levels, and can create problems while training models. For example, you may have artists in your test set that did not appear in your training set. Please avoid such predictors. You may instead collapse a categorical feature with very many levels into one or more categorical feature with fewer levels (e.g. decade of birth of the artist or gender of the artist).
- **Class imbalance.** For classification problems, be wary of class imbalance. You don't need to avoid class imbalance but you must treat such problems with care. Use the suggestions made in Unit 2 Lecture 4, and/or consult the instructor if you have an imbalanced classification problem.