Water Supply Forecasting: San Luis Reservoir

Quang Tran, Darien Young, Nghi Nguyen, and Meishan Fan

Department of Applied Data Science, San Jose State University

DATA 270 Data Analytics Processes

Dr. Eduardo Chan

May 17th, 2022

Abstract

The aim for this project is to leverage machine learning techniques to predict reservoir storage levels at San Luis Reservoir located in California east of Gilroy. San Luis Reservoir is chosen because it provides most of the imported water to Santa Clara County. Using similar research in other hydrological areas, we replicated and adjusted their machine learning models to handle data we procure for Santa Clara County water supply chain. Research on forecasting reservoir storage levels utilized a combination, but not limited to, of input data parameters such as meteorological, snowpack, and runoff. The retrieved data consists of date, precipitation, temperature, snowmelt, and reservoir data because they are the main reservoir inputs for California. XGBoost, Random Forest (RF), Gradient Boost (GB), and Support Vector Regression (SVR) are the four machine learning algorithms used to perform predictions on reservoir storage levels because of their ability to handle limited data and were found to be the best performing repeatedly. The four machine learning algorithms chosen showed high success rates in other research studies utilizing an 80/20 training test split. All of the machine learning models were evaluated with coefficient of determination score (R^2). RF, GB, and XGBoost used root mean square error ($RMSE$), and mean square error (MSE) while SVR utilized mean absolute error (MAE) and mean error (ME). XGBoost, RF, and GB produce similar accuracy but underperform with the data used. SVR using only historical water storage and snow water equivalent data as its two main features produces the highest accuracy rate. The project showed the potential that machine learning can be an additional tool Santa Clara County can use to forecast water supply and push for more localization of experiments done in other areas.

1. Introduction

1.1 Project Background and Executive Summary

In every product sold to consumers, there is a supply chain behind that product that supports the production and the transportation to consumers. Every supply chain has multiple components to it. With so many supply chain choices available, we decided to focus on a part of the water supply.

Water is a vital component to many different supply-chains and production lines, and more often than not it is one of the main limiting resources that forecasters must consider. As the intense level of climate change increases, water supply has become increasingly volatile in the western United States. California, one of the most important agricultural states that provide vegetables, fruits and nuts to countless people in the United States (Pathak, 2018), its water management and planning in the area has also become extraordinarily important. An ample supply of water within California is seasonal, and as a result, water becomes a limited resource when it is out of season. Therefore, the project scope is limited to California.

Leveraging machine learning techniques to forecast the water supply has gained its popularity among water management teams, but there is not one done yet for Santa Clara County. San Luis Reservoir, located in the east of Gilroy and west of Los Baños California, acts as a central water hub for Santa Clara County to collect rainfall from its local area, imported water from the San Joaquin Delta, and by derivative of the Sierra snowpack in California. It distributes part of its stored water to cities throughout Santa Clara County, “Imported water meets about 55 percent of Santa Clara County’s water needs. Most of that (about 40 percent in an average year) is conveyed to the county through the Sacramento-San Joaquin Delta via the State Water Project and Central Valley Project” (“Imported Water”; “South Bay Water Officials”,

2021) and makes up most of the water for Santa Clara County. Hence, the targeted problem of this model development is to predict the water storage level for San Luis Reservoir and the target value is its water storage level. The motivation that helps us determine the target problem comes down to two main reasons. The first reason is that the Santa Clara County area does not have any machine learning models created to address predictions of reservoir storage amount. The second reason is that water is extremely useful in all areas of a society ranging from household chores and everyday usage to industrial production. After reading through different research papers that share similar targeted problems we found that XGBoosting, Gradient Boosting, Random Forest, and Support Vector Regression are the four ML algorithms that are compatible for this project. A preliminary assessment of which model to pick for our project was done by looking for an efficient means of predicting our water supply data, utilizing the data retrieved for this project.

Upon completion of the project, the water storage levels predictions of San Luis Reservoir by different ML models are delivered, which allows further forecasting on water supply allocation to different water retailers in Santa Clara County. This also helps depict the water rate in the area. Water rate, used as a measurement to define the cost of water per square foot, is directly proportional to the water being delivered to consumers and adjusted based on water supply. Many consumers who do not work in the water industry pay more attention to the water rate measurement than measurements such as gallons per minute. How the water rate can be determined more effectively by the water storage levels predictions provide more values on our project. Furthermore, the predictive results may help water consumers plan ahead of time on their water consumption so that water cutting measures are implemented and their lives are not severely affected.

The models come in two phases, the first phase being the code running and the second having results within the scope of our goals. Our goal is to have the predictive results to be as close as possible to the known factual data. An example of this is the predictions of reservoir storage levels for 2019 with historical water storage level data. The collected and verified data from 2019 can be a reference to determine the accuracy of the predictive results. Our development application is essential and the programming scripts are saved in a Jupyter Notebook file.

1.2 Project Requirements

The requirements in this project are (a) SQLite Database which is used to store our data; (b) Python and its associated library packages that runs the models; (c) Google Colab, a platform where models are run on; and (d) Google Drive, where all files and documentations are stored at. Data is retrieved from CSV files pulled from open source data sites. These CSV files are compiled and put into a database to create a single source of truth. Various Python library packages are used for data mining, transformation, and modeling such as Numpy andPandas, for data engineering, Scikit Learn and XGBoost for modeling, Matplotlib and Seaborn for data visualizations. Our database structure also allows us to leverage different machine learning algorithms in a quick manner. In order to create a model for forecasting water supply, we utilize datasets that consist of rainfall, snowfall, snowpacks, temperature (to measure evapotranspiration), and historical water delivery to Santa Clara County. For accurately predicting customer demand we also need to retrieve costs per gallon of water to get an accurate estimate of the water pricing as it varies per year. The testing requirements are to utilize historical data with the results referencing a test year containing ground truth data. Comparing the predicted values to our ground truth values evaluates the accuracy of our model.

1.3 Project Deliverables

Our first deliverable is a detailed project execution plan with the breakdown of subsequent phases using CRISP-DM process model, which is viewed on the Jira platform. . Our roles are planned to be relatively equal and the tasks are assigned based on the individual's availability. This set-up allows us to quickly pivot and compensate if an unplanned emergency occurs in the process of working on this project.

The second deliverable is a detailed exploration report that provides insights into the datasets. The report, for the second deliverable, is planned to be in the form of a Jupyter notebook file with the work being classified under the Data Preparation phase. It includes statistics summaries, graphs, and plots that present factors such as row and column count, percentage of null values, aggregation statistics, and other important details that come to evaluating the health of the dataset.

For our prototype, we expect to have a full framework of each ML algorithm. The predictive results plus the evaluation on the four ML algorithms are on Jupyter Notebook files.

Our deliverables are consistent with the assignment due dates given by Data 270. Some deliverables are subcategories of the main deliverables mentioned in the earlier paragraphs. Table 1 provides a clear outlook on deliverables and their expected completion dates.

Table 1

Overview of Project Deliverables and Dates

Deliverables	Finish Date	Due Date
Assignment 1: Project Proposal	February 17, 2022	February 18, 2022
Chapter 1: Introduction	March 4, 2022	March 5, 2022
Assignment 2: PM Tool & WBS	March 6, 2022	March 7, 2022
Assignment 3: Effort Estimates & Gantt Chart	March 12, 2022	March 13, 2022
Assignment 4: PERT Chart	March 17, 2022	March 18, 2022
Chapter 2: Data & Project Management Plan	March 29, 2022	March 30, 2022
Assignment 5: Data Engineering	April 7, 2022	April 8, 2022
Assignment 6: Data Collection & Data Exploration Plans	April 16, 2022	April 17, 2022
Chapter 3: Data Engineering	April 24, 2022	April 25, 2022
Chapter 4: Model Development	May 4, 2022	May 6, 2022

1.4 Technology and Solution Survey

There were many studies and projects dealing with the water supply chain, but they seldomly covered every part. Often these endeavors cover parts of the water supply chain. A water supply chain could be broken up into five major parts: precipitation, evapotranspiration, water storage, and water demand. From the research done in predicting daily evapotranspiration, Ponraj and Vigneswaran (2019) had datasets “trained, validated, and tested using multiple linear regression, random forest, and gradient boost regression (GBR) algorithms” (p. 5732). In their

research they found that gradient boosting provided the greatest accuracy. A different study on the forecasting monthly reservoir inflow based on machine learning techniques did a comparison among “random forests, gradient boosting machine, extreme learning machine, M5-cubist, elastic net, as well as their multi-model ensemble using Bayesian model averaging (BMA), and tested contributions from different input datasets” also found that gradient boosting provided the most accurate results compared to other algorithms (Tian et al., 2021, p. 1).

Looking at water demand, Shuang and Zhao (2021) research on machine learning predicting water demand had “results suggest that the gradient boosting decision tree (GBDT) model demonstrates the best prediction performance” with testing done in “three other regions in China, and its robustness was validated” (p. 1). Precipitation was a volatile parameter to measure and the research seemed to show that the difference was negligible between the different ML algorithms. Shen and Yong (2021) research did show that “GBDT exhibits more robust features in downscaling the satellite precipitation retrievals than RF over complex terrains, where the amount of precipitation has strong spatial heterogeneity” (p. 1). Through our survey, we found that Gradient Boosting is recommended for predicting almost all aspects of the water supply chain cycle.

In a study done by Nhu et al. (2020), they utilized several “decision tree-based algorithms, including M5 pruned (M5P), Random Forest (RF), Random Tree (RT), and Reduced Error Pruning Tree (REPT)” to predict reservoir water levels.(p. 1) They discovered that the M5P algorithm outperformed all other algorithms in their study, and “works based on regression tasks that have very high dimensionality”(Nhu et al., 2020, p. 4). Based on the research done by Nhu, one of the major limitations of MP5 is that its effectiveness decreases with the lower the dimensionality of a data becomes. In which case Random Forest became the next reasonable

choice as it still performed almost as well as MP5 as shown in the research paper done by Nhu and their fellow researchers.

In another study done by Wang and Wang (2020) where they tested Gaussian process, multiple linear regression, multilayer perceptron, M5P model tree, random forest, and k-nearest neighbors against established prediction model systems such as the "advanced hydrologic prediction system (AHPS)". (p. 1) They found that decision-tree regression machine learning algorithms such as Random Forest all had extremely high precision accuracy in forecasting lake water levels.

Support Vector Machine specific to Support Vector Regression is another machine learning algorithm used as a forecasting model in "reservoir planning and management critical to the development of the hydrological field and necessary to Integrated Water Resources Management." (Hipni, 2013) In an experiment, SVM was used to determine the best time lag to predict dam water levels. Hipni employed "the rainfall $R_{(t-i)}$ and the dam water level $L_{(t-i)}$" (2013) as the best input scenario to get the best time lag, where t represents date and i represents i days before date t.

Another machine learning algorithm compared to SVM is Xgboost. It was also used to forecast water supply. In Malaysia, Xgboost was used to predict groundwater levels in Malaysia in addition to Artificial Neural Network and Support Vector Regression. "Xgboost model outperformed both the Artificial Neural Network and Support Vector Regression models for all different input combinations." (Ibrahem Ahmed Osman, 2020) The input combination being how many days there were in the delay of rainfall data. The best combination of input data was the one that combined rainfall data with the delay of 1,2, and 3 days. The Xgboost found a R^2 of 0.92 which is a great result (Ibrahem Ahmed Osman, 2020)

1.5 Literature Survey of Existing Research

Hussein el at. (2020) employed different machine learning methods such as multilayer perceptron, multivariate linear regression, Random Forest, Extreme Gradient Boosting, and SVR to predict groundwater availability. The process is to conduct the project consisting of using a regression model to predict the groundwater amount, applying feature selection, and finally employing different machine learning approaches (Hussein el at., 2020). As a result, the authors figured out that SVR performs the best when calculating the mean absolute error and root mean squared error.

In two scenarios: interpolation and extrapolation, Shuang and Zhao (2021) proposed Gradient Boosting as the best prediction model for water demand in Beijing-Tianjin-Hebei area in China after testing 11 different machine learning and statistical models. The authors also mentioned that the model then is validated in three other regions of China, and the performance of Gradient Boosting is reconfirmed. The independent variables were associated with the economy, community, water use, and other available resources in the study area (Shuang & Zhao, 2021).

Gradient Boosting was also found to perform strongly by Tian et al. (2021) "forecasting monthly reservoir inflow based on machine learning techniques with dynamic climate forecasts, satellite-based data, and climate phenomenon information" (p.1) . He forecasted two reservoirs: Harris reservoir in humid Alabama and Navajo reservoir in Colorado. Navajo reservoir was the most similar to California's reservoir as its inflow depended on snowmelt runoff in addition to precipitation runoff. Tian et al. (2021) found that the "the gradient boosting machine model with all variables combined as input showed the best performance (KGE = 0.76, R = 0.83)." (p.1)

Ibrahem Ahmed Osman et al. (2021) constructed some predictive models using Extreme Gradient Boosting (XGBoost), Artificial Neural Network (ANN), and Support Vector Regression (SVR) to predict groundwater levels in Selangor, Malaysia. The parameters included the historical data of temperature, rainfall, and evaporation (Ibrahem Ahmed Osman, A et al., 2021). The result was that the XGBoost model outperformed the other machine learning models regardless of all different input combinations, which highlights the potential of the XGBoost algorithm in future groundwater or water supply prediction (Ibrahem Ahmed Osman, A et al., 2021).

Nhu et al. (2020) utilized four decision tree-based algorithms to "predict the daily water level of Zrebar Lake in Iran". The four models utilized in this report are the M5 (pruned) known as M5P, random forest, random tree, and reduced error pruning tree. They tested various combinations of antecedent water level. They used five different lag times to test each of the models to see how well each model was able to predict the water level of the lake. What they found was that out of the four decision tree regression algorithms used, M5P had the highest accuracy followed closely by random forest by about one to two percent on average in each test done. They did outline that the M5P is only effective when the data has a high dimensionality factor to it.

Sapitang et al. (2020) proposed four different models to "forecast changes in water level of a reservoir located in Malaysia". (p. 1) The four algorithms utilized in this research article are decision tree regression, decision forest regression, bayesian linear regression, and neural network regression. The variables utilized in this modeling were water level, rainfall, temperatures, evaporation, and discharge. Sapitang et al. (2020) states that the variables used in this research were important to "forecasting water level" as that was vital in the controlling of

water release during seasonal climate changes. The results that they found were that the Bayesian decision tree regression model had the highest coefficient of determination score out of all of the other models.

Wang and Wang (2020) proposed multiple algorithms and models consisting of "Gaussian process (GP), multiple linear regression (MLR), multilayer perceptron (MLP), M5P model tree, random forest (RF), and k-nearest neighbor" to predict Lake Erie's water level. (p. 1) The goal of this paper was to evaluate the effectiveness of the algorithms and models stated above against the AHPS currently in place. The variables they utilized in their model testing consists of air temperature, wind speed, longwave and shortwave radiation, humidity, precipitation, and water level. The results of their testing showed that the M5P model tree had the lowest root mean squared error and mean absolute error but was on par with the multiple perceptron model in the R^2 evaluation. This stayed consistent through each of the five time splits that were tested on all of the models. In their conclusion Wang and Wang (2020) stated "ML models had higher accuracy than AHPS in predicting water level". (p. 12)

A chart is made in Table 2 to show the comparison among the literature surveys. Each research paper found that their machine learning algorithm was the best in terms of performance and accuracy. Performance and accuracy though is highly dependent on the input parameters and their study area.

Table 2

Comparisons of Literature Surveys

Research Paper	Best Algorithm in The Paper	Study Target
Hussein el at. (2020)	Support Vector Regression	Groundwater Level
Osman et al. (2021)	XGboost	Groundwater Level
Nhu et al. (2020)	Random Forest	Lake Water Level
Shuang and Zhao (2021)	Gradient Boosting	Water Demand from Reservoir Level
Sapitang et al. (2020)	Bayesian Decision Tree	Lake Water Level
Wang & Wang (2020)	M5P	Lake Water Level

2. Data and Project Management Plan

2.1 Data Management Plan

The data collected are datasets that fit the parameters for water supply forecasting commonly used by hydrological researchers and water forecasters. The format of the data pulled is CSV format. This format enables data sharing across different software platforms and data analysis software because of the open-source nature of CSV files and the high popularity of the format.

The datasets collected are from open source and have explicit exceptions for usage in academia purposes. This is achieved by the fact that most of our data sources are public government agencies that release the data explicitly intended for public distribution. Different datasets consisting of specific parameters are necessary for our project. We collected data up to April 2022. The final goal for our project if time permits is to build scraping tools through

Python, or build an automated system to schedule data pulling at the date each dataset gets updated. The schedule is determined by the data collector. This presents an issue on how to handle versioning and storing the data. The manual method of data versioning by changing the file names is inefficient and full of opportunities for errors. In addition, it is not an efficient method of retaining archived versions. To address this, our group utilizes Google Drive as a file hoster and management system. Google Drive cloud functionality allows all group members to work on the same file and see changes in real-time. Google Drive also has functionality for versioning that addresses all of our concerns regarding versioning control. When uploading a file with the same name and format as an existing file on Google Drive, Google Drive automatically knows to replace the file and label it as a new version. Google Drive also allows users to download or revert the previous versions if something went wrong with newer versions. There is no issue regarding storage or limited functionality because Google Drive is tied to SJSU licensing which allows for free and unlimited usage. The retrieved CSV files contain one record per day up to the most recent date pulled. Our datasets should not go beyond a few megabytes.

Google Drive is being used to store our raw and source datasets, but is not a platform we use to store our processed data. We do an ELT Extract, Load, and Transformation (ELT) on all of our raw datasets and move them into a SQLite Database. Using a SQLite Database would make it easier to do data engineering and data preparation as all of the data are located in a single location and can easily be accessed through our data preparation tools. The actual SQLite Database files are hosted on Google Drive to leverage the hosting and versioning capabilities stated in the previous paragraph as the database file is considered a “source” file.

After processing the data to make it standardized so that each dataset can work collaboratively, we leverage research done by other papers to decide on which parameters we use

for our project. Many of the papers we found are very similar and comparable to our project with some minor differences such as different factors they considered in their water resource or different location. In our research of California's water supply chain, we understand that California is a mediterranean climate meaning wet winters and dry summers. This puts a greater emphasis on snow water equivalent and rain run-off. Similar research experiments provide us with better insight and more efficient ways to leverage the parameters that were chosen. The benefits from other research experiments are derived from their experience and knowledge. After deciding on the target parameters we pulled the data from the SQLite database and utilized the data in our four models. Because of the nature of the SJSU license with Google Drive, we plan for this data to be preserved until the end of our Masters program. We feel this is sufficient for the purpose of this project because we can still formulate a new data management plan if preservation is needed beyond.

The data itself is shared using Google’s invitation tool. None of the data or files used and uploaded are on any form of public portal and we do not see any benefits or needs to have this data accessible to the public.An invitation-only is sufficient for the scope of the project.

2.2 Project Development Methodology

Cross Industry Standard Process for Data Mining (CRISP-DM) is the main development methodology for this project, specifically the waterfall methodology of CRISP-DM. There are six phases of CRISP-DM which are: business understanding, data understanding, data preparation, modeling, evaluation, and deployment.

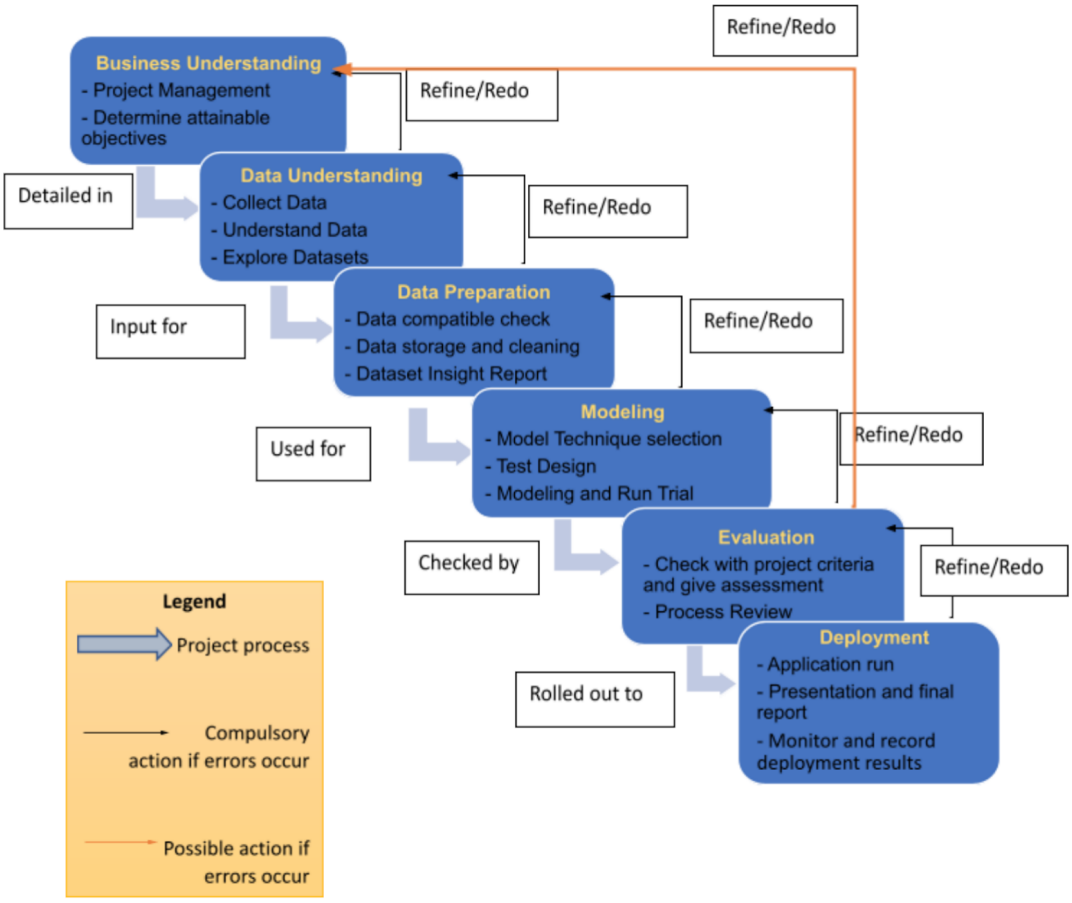
In the Waterfall model, the stages follow a chronological process. We develop a project management plan with the assumption that all project requirements can be understood up front.

The combination of CRISP-DM and Waterfall methodologies results in a horizontal span crossing each phase. A pure Waterfall technique does not allow developers to go back to the previous steps, while a Modified Waterfall Model allows us to return to a previous phase after getting the feedback from the current phase. The process is that we go over tasks in one phase and then move on to the tasks in the next phase. CRISP-DM Agile is a model that focuses on vertical slicing. After completing the first task within business understanding, it is required to go through all the remaining phases to make sure that they are compatible with the tasks. Because of this reason, the CRISP-DM Modified Waterfall approach is the ideal project development methodology. With this project we aim to complete each and every stage before deciding to work on the next stage.

Figure 1 shows the life cycle of our project, project development processes, and activities of our model.

Figure 1

CRISP-DM Modified Waterfall Life Cycle



Note. The thick blue arrows present the chronological process of our project in which the phase at the arrowhead depends on the phase at the tail. Each stage has a black roll-back arrow to the stage before it, meaning that if an error is discovered in any stage then the previous stage must be reviewed, which results in adjustments occurring in the previous stage to ensure the error is fixed. The orange arrow pointing from evaluation to business understanding illustrates a

non-compulsory step if errors occur. Depending on the cause and scale of the errors we may be required to go back to the first stage from the evaluation stage to address the root of the issue.

The project begins with Business Understanding. This phase includes the collection of background information from a variety of sources such as books, research papers, and newspapers. A project management plan is built based on the background information, the project objectives, the success criteria, and a specific project topic. Data requirements, clarifying the problem, and plan to solve the problem decided at this stage after a project topic is figured out.

Data Understanding requires acquiring the data needed for the project based on the topic proposed in the previous phase. The main tasks are collecting the data, understanding the data, verifying, and exploring the data. First, the collecting data task gets the initial data which might include data purchases if needed. Second, it is important to understand the quantitative and qualitative description of the data, such as availability, condition, value types, and coding schemes. Lastly, for the data exploration task, deeper analysis is conducted with the data visualization tools. The analysis can address the data mining goals obtained from the Business Understanding phase. Data Understanding is a critical step as it helps avoid unexpected problems in the Data Preparation step.

The data found in the Data Understanding phase are the inputs for the Data Preparation phase. In Data Preparation, different datasets found in Data Understanding are checked to make sure they are compatible with each other. The data then is stored in a database management system, which is SQLite for our project. At this point, the data goes through the ETL process to handle unnecessary values, null values, outliers, and other data inconsistencies. Data integration among the different datasets is completed after data cleaning . Integration reveals any missed

discrepancies between datasets such as different measurement scales and frequencies. The data is also used to build the Data Insight Report with the data statistics (e.g., number of rows, numbers of null values, aggregation statistics).

Data Preparation provides clean data for the next phase: Modeling. The most appropriate model is chosen for the project's needs with the consideration of data availability, project goals, and other model requirements (e.g. size of the data). This stage also includes a test design creation which should describe the criteria for the accuracy of a model. After that, one or more models are built. Running trial tests on the initial models helps determine the most effective model for the next step, Evaluation.

The final models obtained in the Modeling stage are then evaluated in the Evaluation stage. The Evaluation phase deals with the analysis and assessment of whether the model meets the project criteria. In this stage we would highlight any important and unique findings. We determined the criteria in Business Understanding which makes a correlation between Business Understanding and Evaluation phases. The review process goes through all previous phases and outlines the advantages and disadvantages of each phase for future projects.

The result from the Evaluation step is rolled out to the final step: Deployment. A thorough review is conducted on the entire project. The project implementation is done on a small or local scale. An example of this is running the Python code on a local desktop. If there is no problem, the findings and analysis in the previous stages are condensed into a final report and presented to the class. Deployment results need to be monitored and recorded as well.

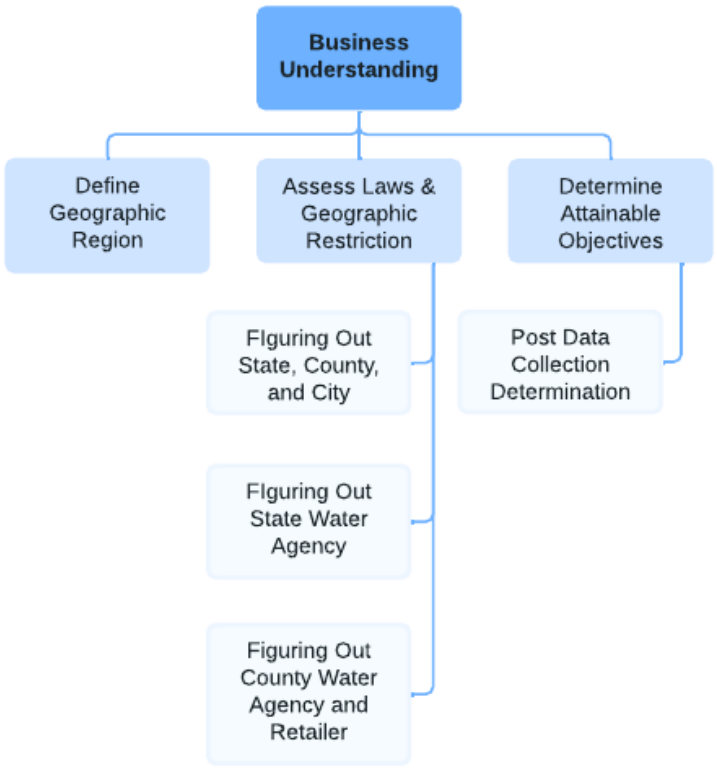
2.3 Project Organization Plan

Having an effective work breakdown structure (WBS) not only makes the project more manageable but also helps teams progress in a project more smoothly. Therefore, our team creates a process-oriented work breakdown structure to help us accomplish this project.

In the first phase of Business Understanding, there are three major outcomes as shown in Figure 2. We first determine which region we would focus on. After we know what region we focus on, we assess the laws and geographic restrictions on water management in that region at both state and county levels. Next, we determine attainable objectives in water forecasting and the final determination on objectives happens after we collect data.

Figure 2

Work Breakdown Structure for Business Understanding Phase

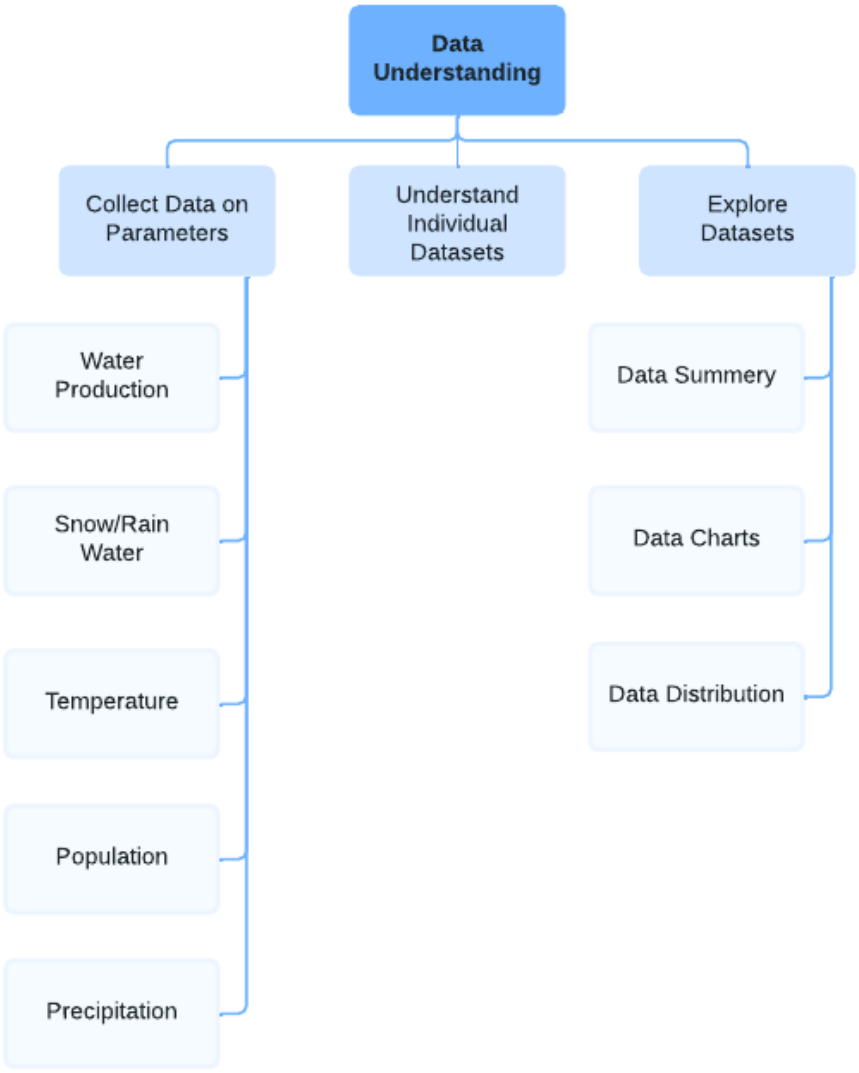


Based on the topic proposed in the previous phase, we move to the next, Data Understanding. In the Data Understanding phase as shown in Figure 3, there are three major tasks: collecting the data, understanding the data, and exploring the data. Collecting water related data being the first task in this phase includes parameters such as production, snow and rain water, temperature, population, and precipitation. After collecting the data, we understand them such as understanding what each column in the datasets meant, how fields are related to each other, and how we can utilize them in our model, etc. Exploring datasets being the last but not the least task in this phase provides us a deeper understanding of the data we collected and this

includes creating data summaries and charts, figuring out how data in each dataset is distributed, and how the data essentially connects to our objectives.

Figure 3

Work Breakdown Structure for Data Understanding Phase

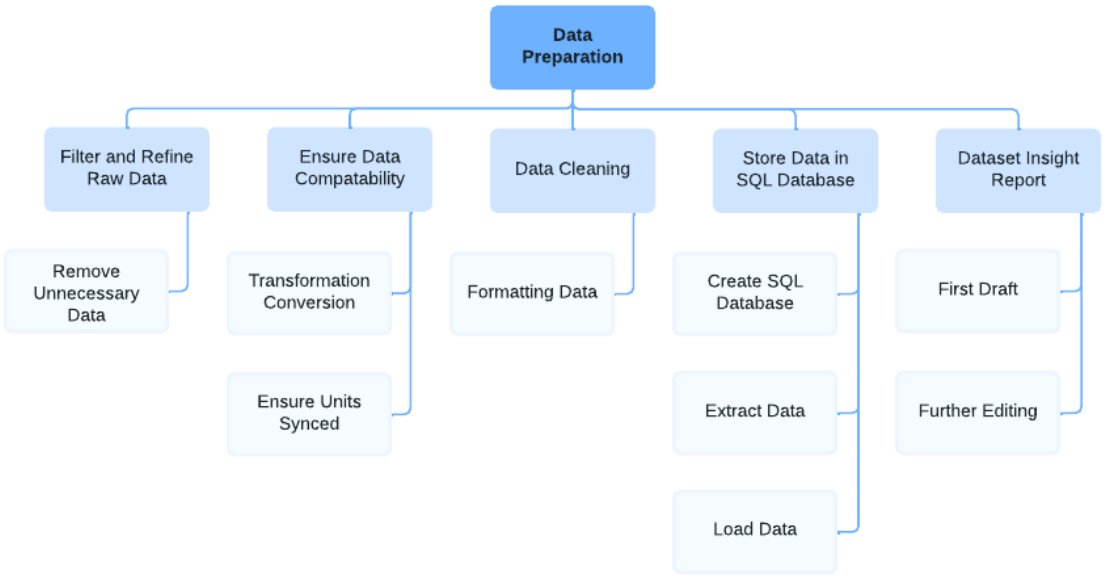


Because a lot of the time the data we initially collect is not clean enough to be used directly, we perform some cleanings to get it ready for our model and that leads to the next phase

of the project, Data Preparation. In the Data Preparation phase as shown in Figure 4, we perform Extract Load and Transform (ELT) on our data. We first store data in SQL database and to accomplish this we first create SQL database, extract data, and load data to the SQL database. Then we perform data filtering such as removing unnecessary data for datasets and data cleaning such as formatting data. After we clean data, we make sure all data in the database is compatible with each other such as ensuring units of measure being all the same and synced, and to achieve this, some transformations and conversions may be required.

Figure 4

Work Breakdown Structure for Data Preparation Phase

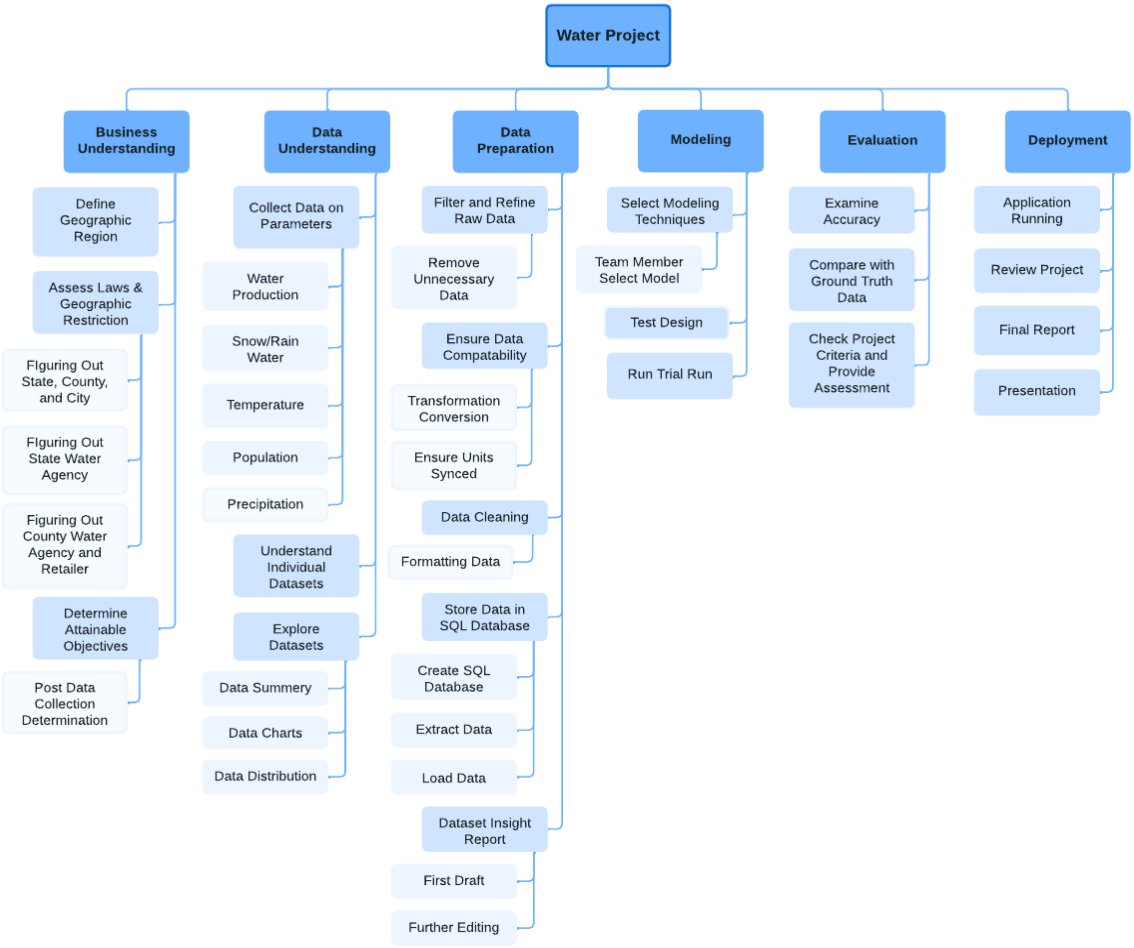


After we have all the data ready, we start the next phase, Modeling. At this stage each team member creates his/her own model, therefore we need to select a modeling technique. Then we design, implement, and evaluate the model. During the evaluation of the models we ensure a desired accuracy has been achieved and we compare our computed data with the ground truth data to examine the accuracy of our model. Lastly, we deploy the model and that would include

application running, review and presentation, and writing a report for the project with our completed WBS as shown in Figure 5.

Figure 5

Work Breakdown Structure for the Project



2.4 Project Resource Requirements and Plan

Hardware and software being utilized consist of software given to us by San Jose State University, free licenses, and our own hardware. There are no costs associated with our project as San Jose State University provides Microsoft Suite, Zoom, and LucidChart academic licenses to students at no cost to the students. We utilize JIRA's free tier license which allows free usage of the app as long as the team is less than 10 people. The open-source software we utilize are Python and SQLite. The data gathered is from open-source datasets coming from government agencies. . JIRA is utilized for project management and Lucidchart for visualizing different charts. Zoom is used for team meetings and SQLite is used as the database management system. Python and its various library packages, such as Pandas, NumPy, Scikit Learn, etc., are used for data manipulation, cleaning, and modeling.We are utilizing our own hardware and free access softwares which does not require us to have an expense. Table 3 below is a summary of all the hardware and software requirements of our project.

Table 3

Resource Requirements & Plan Summary

Function	Resource Type	Resource	Time Duration	Cost
Project Management	Software	JIRA, Lucid Chart	03/04/22 - 03/18/22 (14 days)	Free
Retrieving Data from Data Sources	Software	Microsoft Excel	03/05/22 - 03/26/22 (21 days)	Free
Exploring Data	Software	Microsoft Excel, Microsoft Word	03/22/22 - 03/24/22 (2 days)	Free
Loading Data Into Database	Software	SQLite	03/25/22 - 03/27/22 (2 days)	Free
Data Cleaning	Software	Python, Microsoft Word	03/26/22 - 04/02/22 (7 days)	Free
Insight Report	Software	Python, Microsoft Word	04/01/22 - 04/08/22 (7 days)	Free
Model Testing	Hardware	Python, Hard Drives, CPU usage	04/07/22 - 04/29/22 (22 days)	Free
Model Evaluation	Hardware	Python, Hard Drives	04/25/22 - 05/08/22 (13 days)	Free

The first function of the project resource requirements and plan is the project management step. JIRA is utilized to organize and plan out our project management and to create a Gant consisting of all tasks and subtasks needed for the completion of the project. LucidChart is utilized to create our WBS and PERT charts for our project. This takes approximately 14 days to complete as it requires planning each step of our project out and ensuring that the software required for this function works as intended.

The second function of the project resource requirements is retrieving data from data sources. In this function, we research, locate, and assess potential datasets and ensure that all data matches project requirements. To achieve these, Microsoft Excel is utilized to read the CSV files being pulled from the data sources. This function takes about 21 days to complete as it includes finding the data, ensuring the data found matches with project requirements, pulling the CSV files containing the data, and understanding the data.

The third function of the project resource requirements is exploring data. In this step, we explore the data found in our data files by utilizing Excel to read the CSV files and Word to document what we discovered within each of the different data files. This step takes two days to complete and another two days to allow us to confirm the datasets work for this project.

The fourth function of the project resource requirements is loading the data into our database. This allows us to consolidate our data into a single source for easier transformations and cleaning of all our data. The database management system that we are utilizing is SQLite, because it is easier to utilize and still contains the capabilities of other database management systems. This takes us about two days to account for any unforeseen problems or issues that might arise during database creation.

The fifth function of the project resource requirements is data cleaning. In this phase, we utilize python to extract data from our database and clean it. The cleaning ensures that all data format is correct and unified and any noisy data is sorted out or removed. This takes seven days to complete to ensure a thorough cleaning process is done on the data. Word is utilized to document and ensure that if any data problems arise down the line there is a way to track the changes

The sixth function of the project resource requirements is the insight report which consists of a thorough review of the data. In this report, we create data summaries, a list of potential data problems and resolutions, and reasoning for any data changes done in the data cleaning phase of the project. Python is utilized for this phase to create the data summaries using Pandas and NumPy libraries. Word is used to document the actions done in this phase of the project. This takes about seven days to complete as we need to ensure that all cleaned data is thoroughly explored and consists of necessary data.

The seventh function of the project resource requirements model testing in which we test different machine learning models on our data to see which model would be the most efficient and accurate to utilize for our project. This stage utilizes Google Drive to store the data retrieved from our created database and Python libraries such as Scikit learn to run different machine learning algorithms. Depending on the models being tested it would require significant CPU usage to fully assess the capabilities of the model for our data. We also compare the predictions to the ground truth data that we have gathered. This takes 22 days to complete to ensure thorough testing has been done for each model.

The last function is the model evaluation in which we evaluate how each model works and compare each model to other models to see which models worked the best for our data. This

involves using Python libraries like Scikit learn to utilize the evaluation and comparison tools within the library. These evaluations are stored on Jupyter Notebook files until we compare the different models. This function also consists of a final presentation and final report. This stage takes about 13 days to complete.

2.5 Project Schedule

Figure 6, Figure 7, and Figure 8 shows the Gantt chart for our project. Figure 6 is an overview of all the tasks we have and Figure 7 and Figure 8 shows the detailed start and completion data of each task. All tasks and subtasks are done and presented in the figures. We experienced delays once but other than that, everything was on track.

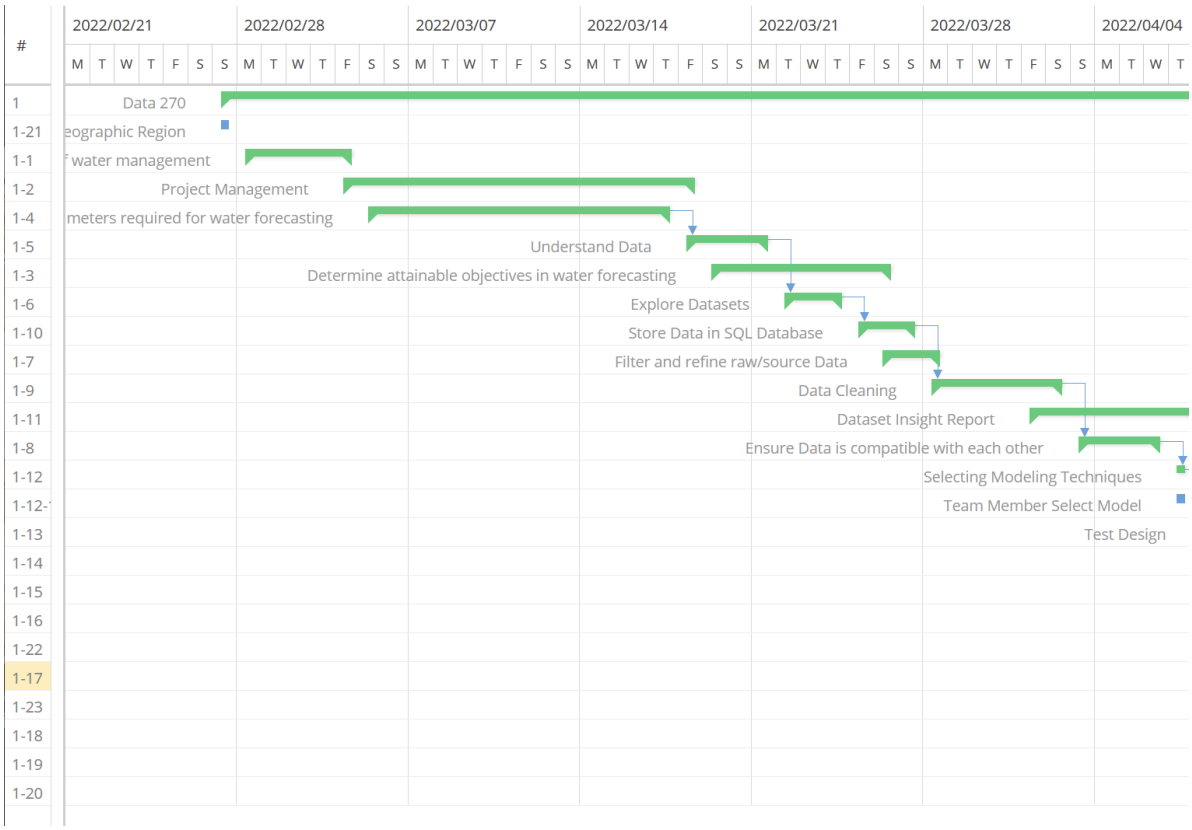
Figure 6

Tasks to Be Completed at Each Stage of The Project

#	Project/Version/Issue	Assignee	Units	Priority	Start	Finish	Durati...	Man...	% Done
1	Data 270				2022/02/27	2022/05/15	78 days		0%
1-21	Define Geographic Region		100%	Me...	2022/02/27	2022/02/27	1 day	No	0%
1-1	Assess the laws and geogr...		100%	Me...	2022/02/28	2022/03/04	5 days	No	0%
1-2	Project Management		100%	Me...	2022/03/04	2022/03/18	15 days	No	0%
1-4	Collect Data on parameter...		100%	Me...	2022/03/05	2022/03/17	13 days	No	0%
1-5	Understand Data		100%	Me...	2022/03/18	2022/03/21	4 days	No	0%
1-3	Determine attainable obje...		100%	Me...	2022/03/19	2022/03/26	8 days	No	0%
1-6	Explore Datasets		100%	Me...	2022/03/22	2022/03/24	3 days	No	0%
1-10	Store Data in SQL Database		100%	Me...	2022/03/25	2022/03/27	3 days	No	0%
1-7	Filter and refine raw/sourc...		100%	Me...	2022/03/26	2022/03/28	3 days	No	0%
1-9	Data Cleaning		100%	Me...	2022/03/28	2022/04/02	6 days	No	0%
1-11	Dataset Insight Report		100%	Me...	2022/04/01	2022/04/08	8 days	No	0%
1-8	Ensure Data is compatible...		100%	Me...	2022/04/03	2022/04/06	4 days	No	0%
1-12	Selecting Modeling Techni...		100%	Me...	2022/04/07	2022/04/07	1 day	No	0%
1-12-1	Team Member Select ...		100%	Me...	2022/04/07	2022/04/07	1 day	No	0%
1-13	Test Design		100%	Me...	2022/04/08	2022/04/16	9 days	No	0%
1-14	Run Trial Run with real data		100%	Me...	2022/04/17	2022/04/24	8 days	No	0%
1-15	Accuracy of Predicted Valu...		100%	Me...	2022/04/25	2022/04/29	5 days	No	0%
1-16	Compare Computed Data ...		100%	Me...	2022/04/25	2022/04/29	5 days	No	0%
1-22	Check with project criteria ...		100%	Me...	2022/04/28	2022/05/01	4 days	No	0%
1-17	Application Running		100%	Me...	2022/04/30	2022/05/01	2 days	No	0%
1-23	Process Review		100%	Me...	2022/04/30	2022/05/03	4 days	No	0%
1-18	Review Whole Project		100%	Me...	2022/05/02	2022/05/03	2 days	No	0%
1-19	Final Report		100%	Me...	2022/05/02	2022/05/15	14 days	No	0%
1-20	Presentation		100%	Me...	2022/05/04	2022/05/08	5 days	No	0%

Figure 7

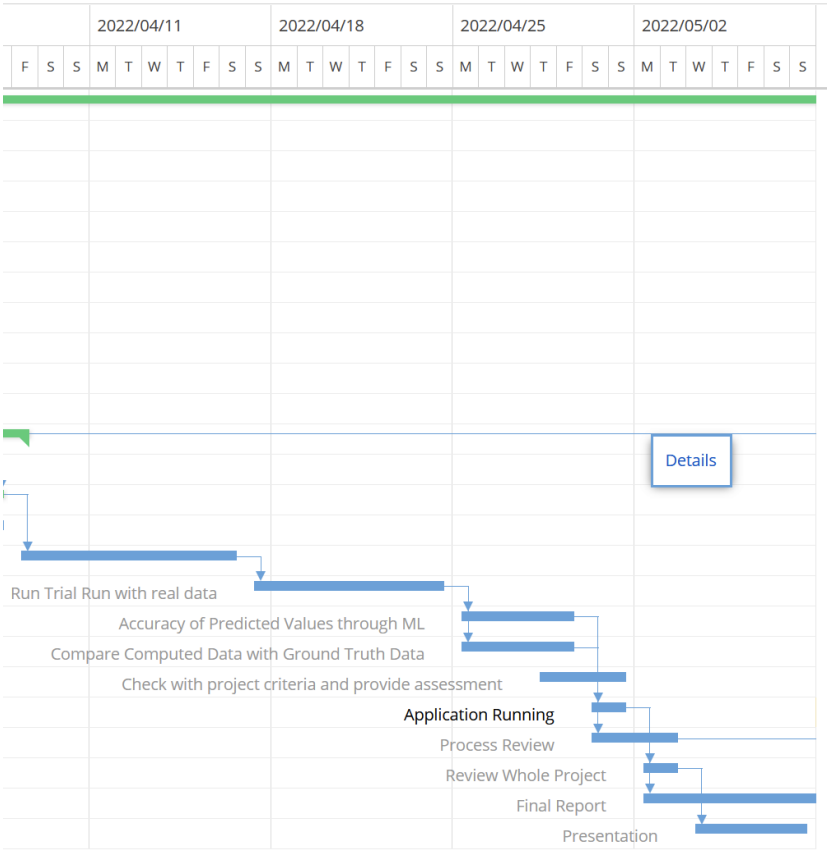
Timeline for Start and Completion of Each Task part 1



Note: The row ID numbers match the row ID numbers in Figure 6

Figure 8

Timeline for Start and Completion of Each Task part 2



Note: The row ID numbers match the row ID numbers in Figure 6

The data understanding phase started on March 5th and was completed by March 24th. All of the data collection tasks have been split up between all four members to collect water production, snow water, population, temperature, and precipitation data. The data collection tasks started on March 5th and are completed by March 17th. The data exploration tasks within this phase were completed by Meishan and Nghi by March 24th. All other tasks within this phase were completed by all four members of the team.

The data preparation phase started on March 22nd and was completed by April 2nd. Storing all of the data within our SQLite database was done by Quang which was completed by

March 27th. The data insight report task was started by all four members concurrently on March 28th until completion on April 8th. The data insight report was a non-critical task and therefore does not need to be completed by April 2nd. The data cleaning and formatting task was completed by Darien by April 2nd.

Our modeling phase was worked on independently and started on April 7th and was completed by April 29th. All tasks within this phase were completed individually by all four members and consisted of picking models, testing the design, running trial runs on the data, checking accuracy of the machine learning, and comparing computed data with ground truth data.

The evaluation phase of the project started on April 28th and was completed on May 1st. This phase purely consisted of checking the project criteria and assessing the models to ensure they match the project requirements.

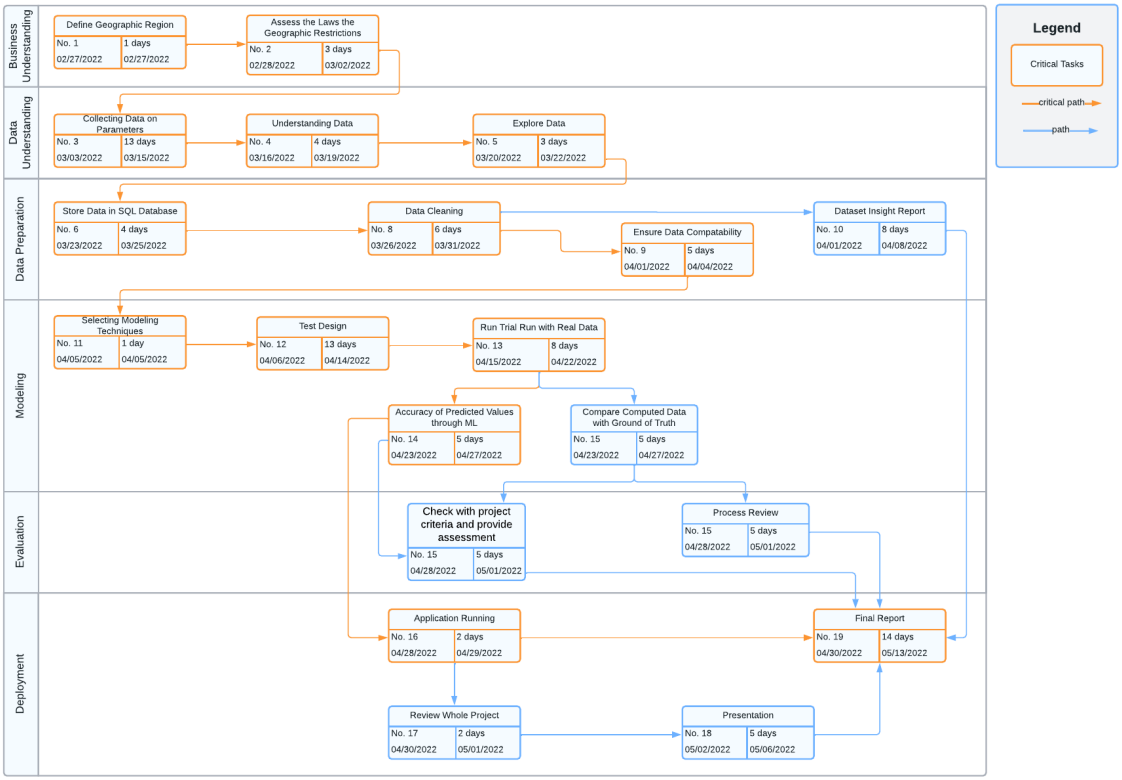
The deployment phase of the project started on April 30th and was completed by May 15th. This phase was worked on by all four members of the team and consisted of ensuring our primary model is running smoothly, reviewing the entire process of the project, writing up a final report, and presenting our findings to the class.

The PERT chart in Figure 9 shows the critical path that are taken to complete our project. It also shows all tasks that are non-critical and do not have as much of an impact on project completion.

Every fork made in the PERT chart and their following tasks only served to verify our assumption that the data is accurate. Though verification was important, the removal or deprioritization of it did not fundamentally affect our project.

Figure 9

PERT Chart for the Project



3. Data Engineering

3.1 Data Process

The first step is to identify the desired attributes for our data, which depends on two main factors: the source has to be trusting and viable, and the datasets must have at least one of the necessary attributes to the targeted problem as described in Section 1.1. The critical attributes for our project are reservoir storage amount, snow melt data, precipitation, and temperature data. In order to satisfy those requirements we collect the data through government agencies such as National Oceanic and Atmospheric Administration (NOAA), United States Department of Agriculture (USDA), and California Natural Resources Agency (CNRA).

For Snow, hydrologists and other water-related forecasters use a measurement called snow water equivalent which determines how much water is produced from snow melt. California is a special case where a significant portion of the water supply relies on snow water. It means the State invests heavily in recording snow packs and there is a lot of snow historical data. California snowpacks are limited to just the Sierra Mountain Range and many organizations rely on a few players to handle the measurement recordings. In this project we rely on the SNOTEL snow sensor network. These are remote sensors which send out reports daily. Specifically, we choose the USDA whose customers heavily rely on accurate data and inaccurate data has a quick turnaround of feedback. This creates an environment where the agency wants to ensure reliable data and quick feedback. Other agencies may have their data seldomly checked which creates an environment where there is lack of quality control. We find snow melt data on the USDA is within a single CSV file which allows for an easy download and loading into our database. After obtaining snow water data we move on to finding weather data.

NOAA containes weather data grouped by weather stations in multiple different CSV files. With this we download all of the weather stations in California. We concentrate on a 50 mile buffer around the San Luis Reservoir as all imported water for Santa Clara County comes from the San Joaquin/Sacramento area and must go through the San Luis reservoir. Because of time limitations, we want to limit the scope of the project, and the San Luis Reservoir is a good checkpoint and target. The datasets from NOAA contain the precipitation and temperature data. Using the Pandas library within Python we merge all of the datasets into one large weather dataset which could be loaded into our database. The next step is retrieving the reservoir target data necessary for our project.

As mentioned earlier, we concentrate our efforts on the data from San Luis reservoir. The

reservoir data is located on the CNRA website which contains data on reservoirs from all over California managed by the State. This does not include reservoirs managed at the local level but in the scope of the project this is not needed. In order to reduce the amount of data that is needed to be cleaned and transformed, we pull only reservoirs in Northern California. From there we merge all of the reservoir data into one CSV file using Pandas for easier loading.

During the data cleaning stage, we make date values a consistent format, clean up nulls, and address any outliers in the datasets. After cleaning, we eliminate unnecessary parameters/columns and perform data aggregation transformation to obtain the required parameters. Next, we combine the required parameters to form a combined table. The combined table is normalized and regularized using LASSO regularization.

Training and testing is conducted on the transformed combined table. We use the 80/20 strategy for training and testing. We do not have a validation set as KFold Cross Validation is used in the model development phase, and it splits the training dataset into sub-training sets and a validation set. This is done by using scikit learn train_test_split definition to split our data into the two categories. This ensures the data we use for each model is the same so that changing data do not have to be factored in when we are comparing models.

3.2 Data Collection

3.2.1 Weather Dataset.

Data Requirements.

This dataset is collected from NOAA, which is an authorized government agency for environmental data, including atmospheric, geophysical, and oceanic research all around the world (National Centers for Environmental Information, 2022). The weather data we withdraw is from Global Historical Climatology Network - Daily dataset on the NOAA website (National

Centers for Environmental Information, 2022).

From this dataset, we need daily precipitation and temperature related data for stations within the 50mi region of San Luis Reservoir. The dataset contains the historical (daily) data from January 1, 2014 to April 4, 2022 for 1814 stations in California. There are 3305060 rows in total. The dataset is collected on a daily basis where each station is recorded once a day. The National Centers for Environmental Information (2022) provides the definition for each parameter as follows.

- Station: Station ID
- Name: Station Name
- Latitude: Station's latitude
- Longitude: Station's longitude
- Date: The date of the record
- PRCP: Daily precipitation amount
- TAVG: Average temperature
- TMAX: Maximum temperature
- TMIN: Minimum temperature

Figure 10 shows a detailed data collection plan for weather data. It shows how we collect the data as well as the characteristics of each variable.

Figure 10

Data Collection Plan for Daily Precipitation and Temperature

Variables	Date	Station ID	Station Name	Latitude	Longitude	Precipitation	Average Temperature	Max Temperature	Min Temperature
Input (X) or output (Y)?	X	X	X	X	X	X	X	X	X
Unit of Measurement	mm/dd/yyyy	N/A	N/A	Degree	Degree	Inches	Fahrenheit (°F)	Fahrenheit (°F)	Fahrenheit (°F)
Data Type	String	String	String	Decimal	Decimal	Decimal	Decimal	Decimal	Decimal
Collection Method	Existing Data Collection								
Historical Data Exist?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Source of Historical Data	National Centers for Environmental Information (NOAA)								
Mean	N/A	N/A	N/A	37.11	-120.23	0.08	56.39	71.12	47.04
Upper Specification Limit	04/04/2022	No Limit	No Limit	42	-114.17	14.53	150	150	150
Lower Specification Limit	01/01/2014	No Limit	No Limit	32.56	-124.34	0	-79	-79	-80
Standard Deviation	N/A	N/A	N/A	2.65	2.27	0.32	15.14	16.97	13.68
Historical data reliable?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Collector	Weather Stations								
Operational Description exist?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Resources available?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Start Date	03/03/2022								
End Date	03/15/2022								
Duration (in days)	13								

Note. For better readability, the table is screenshotted from the original.

Sample raw data.

The head and tail of the sample raw dataset are shown in Figure 11.

Figure 11

Sample Raw Data

	STATION	NAME	LATITUDE	LONGITUDE	DATE	PRCP	TAVG	TMAX	TMIN
0	USR0000CTHO	THOMES CREEK CALIFORNIA, CA US	39.8644	-122.6097	2022-01-01	NaN	41.0	53.0	34.0
1	USR0000CTHO	THOMES CREEK CALIFORNIA, CA US	39.8644	-122.6097	2022-01-02	NaN	37.0	46.0	30.0
2	USR0000CTHO	THOMES CREEK CALIFORNIA, CA US	39.8644	-122.6097	2022-01-03	NaN	42.0	51.0	32.0
3	USR0000CTHO	THOMES CREEK CALIFORNIA, CA US	39.8644	-122.6097	2022-01-04	NaN	48.0	50.0	45.0
4	USR0000CTHO	THOMES CREEK CALIFORNIA, CA US	39.8644	-122.6097	2022-01-05	NaN	49.0	57.0	42.0
...	...	...	...	...	...	...	...	...	...
3305055	USC00046006	MOUNT WILSON CBS, CA US	34.2308	-118.0711	2014-12-27	0.00	NaN	51.0	28.0
3305056	USC00046006	MOUNT WILSON CBS, CA US	34.2308	-118.0711	2014-12-28	0.00	NaN	54.0	36.0
3305057	USC00046006	MOUNT WILSON CBS, CA US	34.2308	-118.0711	2014-12-29	0.00	NaN	58.0	36.0
3305058	USC00046006	MOUNT WILSON CBS, CA US	34.2308	-118.0711	2014-12-30	0.07	NaN	41.0	23.0
3305059	USC00046006	MOUNT WILSON CBS, CA US	34.2308	-118.0711	2014-12-31	0.03	NaN	35.0	18.0

3305060 rows × 9 columns

3.2.2 Snow Dataset.

Data Collection Requirements.

Another parameter that plays an important role in building our machine learning model is snow water equivalent. Snow water equivalent is the measurement used to determine how much water is produced when snowpacks melt. Snow water equivalent data is collected through "automated data collection sites located in remote, high-elevation mountain watersheds in the western US" (United States Department of Agriculture, 2022) called SNOTEL. We grab our raw data source, made up of responses from SNOTEL, from the USDA National Water and Climate Center. The raw dataset contains 402,725 records which equates to 402,725 sensor records obtained from the SNOTEL network. We do not go out to collect the Snow Water Equivalent data ourselves as we do not have the manpower or budget to do so. Instead we accept the data from the USDA as a source of truth and move on. Below is a table summarizing how each

column is handled in our project. Since the snow water equivalent and snow depth have a 1:1 relationship, it is expected that the rate of change between the two is the same. Table 6 below shows a detailed data collection plan for snow water equivalent data. It shows how we collect the data as well as the characteristics of each variable.

Table 6

Data Collection Plan for Snow Water Equivalent

Variables	Date	Station_Name	Station_Id	Snow_Water_Equivalent_(in)_Start_of_Day_Values	Change_In_Snow_Water_Equivalent_(in)	Snow_Depth_(in)_Start_of_Day_Values	Change_In_Snow_Depth_(in)
Input (X) or output (Y) variable?	X	X	X	X	X	X	X
Unit of measurement	mm/DD/yyyy	N/A	N/A	inches	inches	inches	inches
Data type	Date	String	INT	NUM	NUM	NUM	NUM
Collection Method	SNOTEL Sensor Network						
Historical Data exist?	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Source of historical data	USDA Natural National Water and Climate Center						
Mean	N/A	N/A	N/A	6.75	0.005	18.25	0.005
Upper specification limit	N/A	N/A	N/A	122.1	44	280	44
Lower specification limit	N/A	N/A	N/A	0	-17	0	-17
Standard Deviation	N/A	N/A	N/A	12.86	2.36	31.041	2.36
Historical data reliable?	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Collector	SNOTEL Sensor Network						
Start Date	03/03/2022						
End Date	03/15/2022						
Duration (in days)	13						

Note. For better readability, the table is screenshotted from the original.

Sample Raw Data.

Using Jupyter Notebook and Pandas dataframe functionality, we are able to grab a sample of our dataset which highlights the head and tail of our snow dataset. Quick overview, we find that our dataset is organized by “Date” in ascending order and then grouped by Station_ Name which is also in ascending order. Figure 12 shows a sample of the snow water equivalent dataset.

Figure 12

Sample of SNOTEL Dataset

	Date	Station_Name	Station_id	Snow_Water_Equivalent_(in)_Start_of_Day_Values	Change_In_Snow_Water_Equivalent_(in)	Snow_Depth_(in)_Start_of_Day_Values	Change_In_Snow_Depth_(in)
0	10/1/1984	Adin Mtn	301	0			
1	10/2/1984	Adin Mtn	301	0	0		
2	10/3/1984	Adin Mtn	301	0	0		
3	10/4/1984	Adin Mtn	301	0	0		
4	10/5/1984	Adin Mtn	301	0	0		
...	...	...	...	...	...	...	...
402720	4/2/2022	Ward Creek #3	848	17.5	-0.8	31	-1
402721	4/3/2022	Ward Creek #3	848	16.8	-0.7	30	-1
402722	4/4/2022	Ward Creek #3	848	16.4	-0.4	30	0
402723	4/5/2022	Ward Creek #3	848	15.2	-1.2	29	-1
402724	4/6/2022	Ward Creek #3	848	14.5	-0.7	18	-11

402725 rows × 7 columns

3.2.3 Reservoir Dataset.

Data Collection Requirements.

One of the most important parameters needed for our machine learning is the reservoir storage amount. The reservoir storage amount helps to accurately define how much water is available for supplying Santa Clara County. The reservoir storage amount is cataloged and measured by the California Natural Resources Agency (CNRA) in which they utilize tools such as staff gauge markers to measure the height of the water and the amount stored in the reservoir at any given time. The recording and measurement is part of the State Water Project (SWP) which maintains accurate records of California's water resources, systems, and infrastructures. The data is retrieved from Daily SWP Reservoir Elevation and Storage Data located at (https://data.cnra.ca.gov/dataset/). The attributes in the dataset are date, Stage (ft), storage, reservoir, elevation, and field_division. The attribute Stage_(ft) is the water level of water in the reservoir measured by feet. The storage is how much acre-feet of water is being stored in the reservoir with acre-feet being the volume of water that would cover one acre to a depth of one foot. The elevation is how high the water is in the reservoir and field_division is the group responsible for overseeing the reservoir. The raw dataset consists of 856995 rows worth of data

with about 21 different reservoirs with recordings occurring every 15 minutes. With our focus being in the Santa Clara County area we focus on the San Luis reservoir which consists of 15796 rows worth of data. Table 7 summarizes the data attributes and data collection necessary for this dataset.

Table 7

Data Collection Plan for Reservoir Data

Variables	date	Stage (ft)	storage	reservoir	elevation	field_division
Input (X) or output (Y) variable?	X	X	Y	X	X	X
Unit of measurement	N/A	feet	acre-feet	N/A	feet	N/A
Data type	Date	Float	Float	String	Float	String
Collection Method	Data was collected utilizing water level data loggers, staff gage markers, and other telemetry units					
Historical Data exist?	Yes	Yes	Yes	Yes	Yes	Yes
Source of historical data	State Water Project Reservoir					
Mean	N/A	630.44	85619.17	N/A	929.21	N/A
Upper specification limit	N/A	783.6	3578364	N/A	5777.02	N/A
Lower specification limit	N/A	473.5	0	N/A	-3.93	N/A
Standard Deviation	N/A	120.81	371042.7	N/A	1215.07	N/A
Historical data reliable?	Yes	Yes	Yes	Yes	Yes	Yes
Data Collector	California Natural Resources Agency					

Start Date	03/03/2022
End Date	03/15/2022
Duration (in days)	13

Sample Raw Data.

Utilizing Python and the built in pandas dataframe we are able to create a dataframe that shows the head and tail of our raw dataset for reservoir data (Figure 13). Looking at the raw dataset we notice that it is ordered in ascending order by date and reservoir name. Each reservoir has different recorded start dates which could be the result of some reservoirs being older then others or the reservoirs are not being recorded until the very first recorded date. For example Coyote Hills only has reservoir data from 2014 to present day where San Luis has data from 1979 to present. Coyote Hills reservoir has been created recently which is why it only has data going back to 2014.

Figure 13

Sample of State Water Project reservoir dataset

	date	Stage_(ft)	storage	reservoir	elevation	field_division
0	2014-01-01T00:00:00	747.1	7849.19	Coyote Hills	534	
1	2014-01-01T00:15:00	747.1	7849.19	Coyote Hills	534	
2	2014-01-01T00:30:00	747.1	7849.19	Coyote Hills	534	
3	2014-01-01T00:45:00	747.1	7849.19	Coyote Hills	534	
4	2014-01-01T01:00:00	747.1	7849.19	Coyote Hills	534	
...	...	...	...	...	...	...
856990	2022-04-07T17:15:00	507.5	1102.67	Stevens Creek	554	
856991	2022-04-07T17:30:00	507.5	1102.67	Stevens Creek	554	
856992	2022-04-07T17:45:00	507.5	1102.67	Stevens Creek	554	
856993	2022-04-07T18:00:00	507.5	1102.67	Stevens Creek	554	
856994	2022-04-07T18:15:00	507.5	1102.67	Stevens Creek	554	

856995 rows × 6 columns

3.3 Data Pre-processing

3.3.1 Weather Data.

Missing and Incomplete Data.

The weather data we collect has 1808 weather stations all over California. The most important data we care about are Precipitation, Temperature Max, Temperature Minimum, and Average Temperature and since we only want stations within the 50 miles of San Luis Reservoir, we filter out stations and eliminate the ones that are not inside the boundary. There are 95 stations within 50 miles as shown in Figure 14. In total, we have about 170000 rows.

Figure 14

Sample Of Stations That Are Within 50 Miles From The San Luis Reservoir

```
STATION      NAME
US1CAMA0008  CHOWCHILLA 0.3 E, CA US                2338
US1CAME0004  MERCED 1.2 NW, CA US                  2124
US1CAMT0006  CASTROVILLE 1.6 ENE, CA US            2572
US1CAMT0017  AROMAS 2.0 SSW, CA US                  405
US1CAMT0020  CASTROVILLE 1.6 ENE, CA US            2910
                                                   ...
USW00023257  MERCED MUNICIPAL AIRPORT, CA US       3015
USW00023258  MODESTO AIRPORT, CA US                3015
USW00023277  WATSONVILLE MUNICIPAL AIRPORT, CA US  3012
USW00023293  SAN JOSE, CA US                       3015
USW00093243  MERCED 23 WSW, CA US                  2838
Length: 95, dtype: int64
```

Each station should have 3016 days of data (from January 1, 2014 to April 4, 2022). However, among the 95 stations, there are some stations that do not have a lot of days of data recorded. Furthermore, there is one station, Castroville 1.6 ENE, CA, has more days of data recorded than it should. Because having many days of missing data in a station would jeopardize the accuracy of prediction, we eliminate stations that are below a certain completed rate.

To select a completed rate, we chose 90.0% for TMAX, 90.0% for TMIN, and 90.0% for PRCP. As a result, in a total of 3016 days from January 1, 2014 to April 4, 2022, 90.0% of completed rate gives us at least 2714 days of data. In other words, we only have at most 301 days of missing data, meaning in average we have about 3.5 days of missing data per month which should not affect the result of our prediction by much. The selection leaves us six stations listed below.

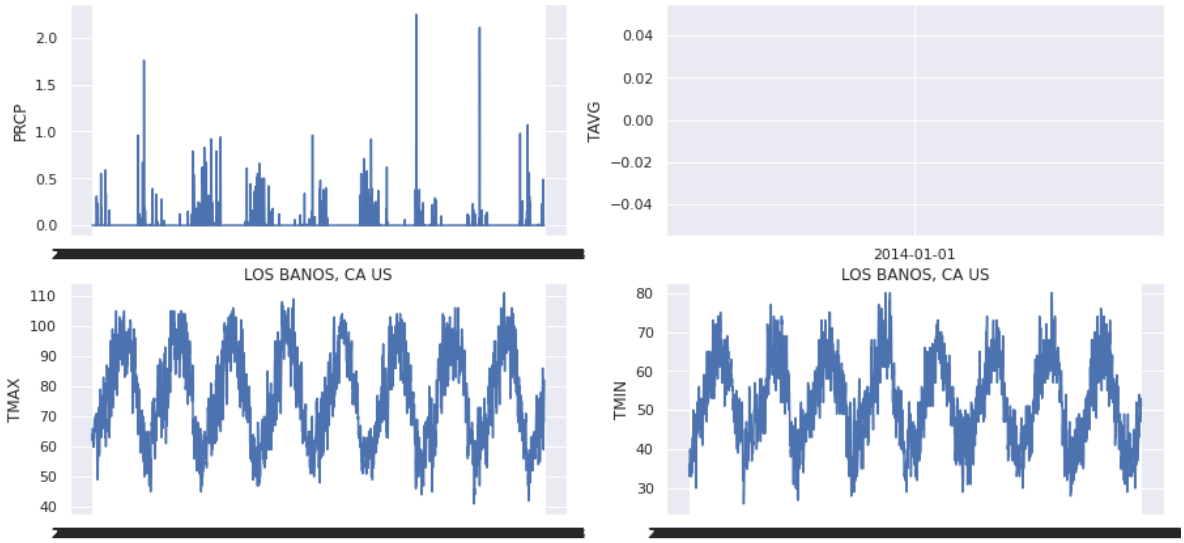
- Los Banos, CA US
- Mount Hamilton, CA US
- Newman, CA US
- Watsonville Waterworks, CA US
- Salinas Airport, CA US
- Merced 23 WSW, CA US

Identify Trends and Distributions

Next, we visualize each station and its parameters including temperature min (TMIN), max (TMAX), average (TAVG), and precipitation (PRCP). We graph line charts for each station. The line charts in Figure 15 show a trend for each parameter of Los Banos Reservoir from January 1, 2014 to April 4, 2022. The charts for TAVG of each station are empty because of the missing values. Below is an example of one of the stations and all other stations appear to have the same patterns.

Figure 15

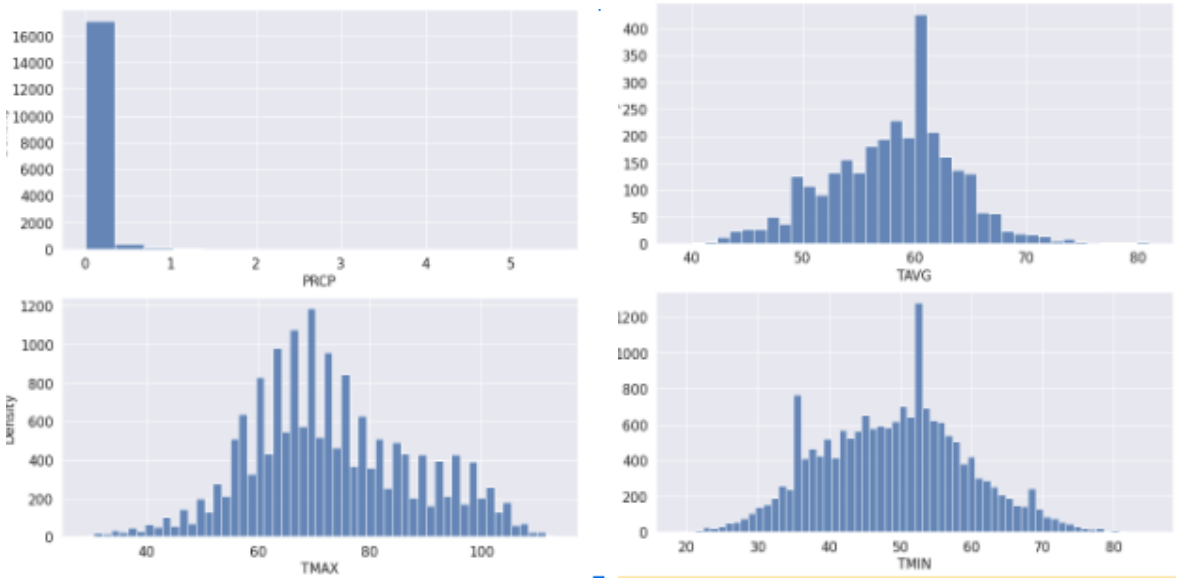
Los Banos, CA US Station and The Trend of Its Four features Covers from January 1, 2014 to April 4, 2022.



According to Figure 16, PRCP is exponentially distributed. TMIN, and TAVG appear to be normally distributed while TMAX seems bimodal distributed.

Figure 16

Los Banos, CA US Station and The Histogram of It Four Features Covers from January 1, 2014 to April 4, 2022.

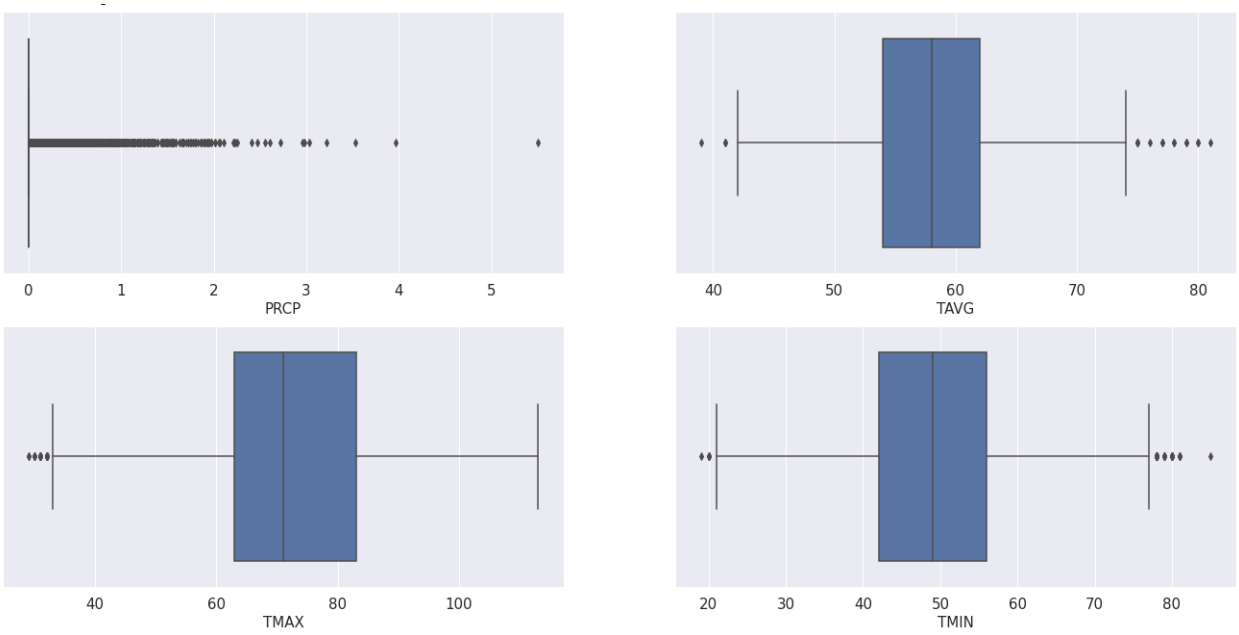


Identify Outliers.

According to the boxplots in Figure 17, there exist outliers but they are all within an acceptable range considering the geographic location of the station.

Figure 17

The Plot Box Distribution of The Six Selected Stations' Four Features



Handling Null and Incomplete Data.

At this point, we check the amount of missing values we have, which is proven to be zero. To deal with the missing values in PRCP, TMAX, and TMIN, we first pivot the weather table with the columns of the six stations and their corresponding features. For example, one column illustrates the precipitation of Los Banos station while another column shows the average temperature of Los Banos station. Then we use a function from the Pandas Python library called interpolate, linear as the method to replace the null values. Linear method in this case should work because the consecutive missing values we have do not exceed 12 and it is proven by running functions that all missing values are replaced. Then what is left to deal with is TAVG. After going over some values of TAVG, we figured that the TAVG values are equal to or very close to the average of TMAX and TMIN. Therefore, we took the average of TMAX and TMIN

of the station of the date as the TAVG value. Figure 18 is a sample of how our table looked after filling out the missing values.

Figure 18

Weather Pivoted Table Sample

	PRCP						TAVG			
STATION	USC00045118	USC00045933	USC00046168	USC00049473	USW00023233	USW00093243	USC00045118	USC00045933	USC00046168	USC00049473
DATE										
2014-01-01	0.0	0.0	0.0	0.0	0.0	0.0	49.0	57.5	50.0	52.50
2014-01-02	0.0	0.0	0.0	0.0	0.0	0.0	49.0	59.5	51.0	54.00
2014-01-03	0.0	0.0	0.0	0.0	0.0	0.0	49.0	58.5	51.0	57.00
2014-01-04	0.0	0.0	0.0	0.0	0.0	0.0	49.0	53.5	51.5	54.00
2014-01-05	0.0	0.0	0.0	0.0	0.0	0.0	49.5	56.0	50.5	57.00
...	...	...	...	...	...	...	...	...	...	...
2022-03-31	0.0	0.0	0.0	0.0	0.0	0.0	57.5	47.0	52.0	53.75
2022-04-01	0.0	0.0	0.0	0.0	0.0	0.0	58.5	53.5	52.0	53.50
2022-04-02	0.0	0.0	0.0	0.0	0.0	0.0	63.0	57.0	52.0	53.50
2022-04-03	0.0	0.0	0.0	0.0	0.0	0.0	65.0	51.0	52.0	55.50
2022-04-04	0.0	0.0	0.0	0.0	0.0	0.0	66.5	51.0	52.0	55.50

3016 rows × 24 columns

3.3.2 Snow Water Equivalent Data.

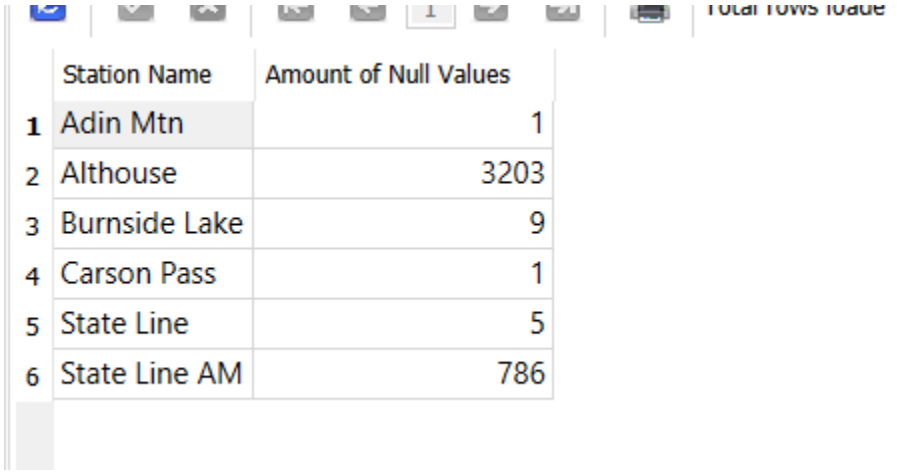
Missing and Incomplete Data.

Since our data is a csv import, everything is a string including the null values. As a result, “IS NULL” command does not work. Searching for null values, we find that there are about 4005 records. This is about 0.9% of our dataset.

With further investigations, only six stations listed in Figure 19 have NULL values. Only two of the stations had significantly high NULL Values. Since we have 36 stations in total, it does not truly corrupt our dataset.

Figure 19

Stations With Their Total Count Of Null Values



	Station Name	Amount of Null Values
1	Adin Mtn	1
2	Althouse	3203
3	Burnside Lake	9
4	Carson Pass	1
5	State Line	5
6	State Line AM	786

Specifically looking at the two stations with the outlier amount of NULL values, we find that the stations have not returned any data in the time period we selected for the dataset.

Our data starts in 2014. There are three stations that did not start the data recording until a few years after 2014. The three stations are , Althouse station, and Fredonyer Peak station, and State Line AM station line number 2, 14, and 31 in Figure 20 respectively Therefore, the total number count of records are especially low. For consistency and since we have enough data from the other stations, we remove Fredonyer Peak to have consistent data where each station has snow water equivalent data

Figure 20

Total Records of SNOTEL Stations

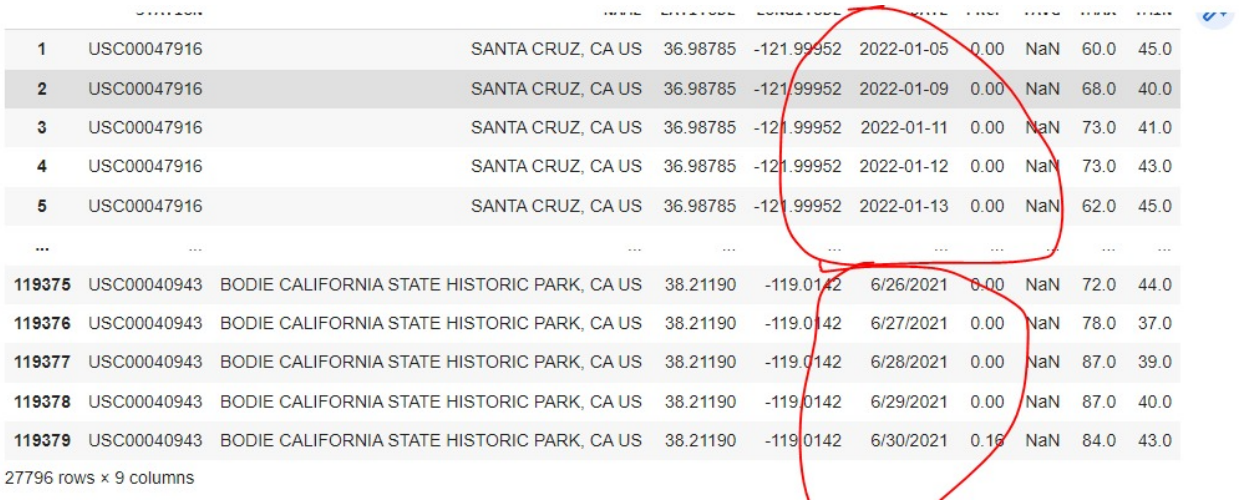
	Station Name	Amount of Recordings
1	Adin Mtn	13701
2	Althouse	3203
3	Blue Lakes	13701
4	Burnside Lake	6772
5	Carson Pass	6398
6	Cedar Pass	13702
7	Crowder Flat	8223
8	Css Lab	13702
9	Dismal Swamp	13702
10	Ebbetts Pass	13702
11	Echo Peak	13702
12	Fallen Leaf	13702
13	Forestdale Creek	6763
14	Fredonyer Peak	1975
15	Hagans Meadow	13702
16	Heavenly Valley	13702
17	Horse Meadow	6763
18	Independence Camp	13702
19	Independence Creek	13702
20	Independence Lake	13702
21	Leavitt Lake	11876
22	Leavitt Meadows	13702
23	Lobdell Lake	13702
24	Monitor Pass	11511
25	Palisades Tahoe	13702
26	Poison Flat	13702
27	Rubicon #2	13702
28	Sonora Pass	13702
29	Spratt Creek	13702
30	State Line	2847
31	State Line AM	786
32	Summit Meadow	6763
33	Tahoe City Cross	13702
34	Truckee #2	13701
35	Virginia Lakes Ridge	13702
36	Ward Creek #3	13702

Inconsistent Data Format.

The data found in the SNOTEL datasets are not consistent in terms of the date format. Some stations utilize mm/dd/yyyy while others utilize yyyy-mm-dd as noted in Figure 21. To resolve this we change all the date formats of yyyy-mm-dd into mm/dd/yyyy.

Figure 21

Mismatched Dates



1	USC00047916	SANTA CRUZ, CA US	36.98785	-121.99952	2022-01-05	0.00	NaN	60.0	45.0
2	USC00047916	SANTA CRUZ, CA US	36.98785	-121.99952	2022-01-09	0.00	NaN	68.0	40.0
3	USC00047916	SANTA CRUZ, CA US	36.98785	-121.99952	2022-01-11	0.00	NaN	73.0	41.0
4	USC00047916	SANTA CRUZ, CA US	36.98785	-121.99952	2022-01-12	0.00	NaN	73.0	43.0
5	USC00047916	SANTA CRUZ, CA US	36.98785	-121.99952	2022-01-13	0.00	NaN	62.0	45.0
...	...	...	...	...	...	...	...	...	...
119375	USC00040943	BODIE CALIFORNIA STATE HISTORIC PARK, CA US	38.21190	-119.0142	6/26/2021	0.00	NaN	72.0	44.0
119376	USC00040943	BODIE CALIFORNIA STATE HISTORIC PARK, CA US	38.21190	-119.0142	6/27/2021	0.00	NaN	78.0	37.0
119377	USC00040943	BODIE CALIFORNIA STATE HISTORIC PARK, CA US	38.21190	-119.0142	6/28/2021	0.00	NaN	87.0	39.0
119378	USC00040943	BODIE CALIFORNIA STATE HISTORIC PARK, CA US	38.21190	-119.0142	6/29/2021	0.00	NaN	87.0	40.0
119379	USC00040943	BODIE CALIFORNIA STATE HISTORIC PARK, CA US	38.21190	-119.0142	6/30/2021	0.16	NaN	84.0	43.0

27796 rows × 9 columns

Identify Outliers.

Identifying outliers requires us to have hands-on knowledge of how snowpacks work in California and to utilize Python scripting to make boxplots to identify outliers. What we found is that there is no outlier data on the lower end spectrum, below zero, which easily highlights an error. On the higher spectrum it is a little more difficult to identify as there are legitimate times when a snowpack increases exponentially such as when a snowstorm happens. In this case the easiest way to identify a true outlier is to relate the rate of change to the time period. With the specific metric determining that higher levels of SWE or snow depth are happening within the wet season rather than the dry season.

3.3.3 Reservoir_data.

Understanding Raw Data Source.

Since our project area is Santa Clara County, we currently only care about one reservoir managed at the state level. San Luis Reservoir located near Los Banos. Filtering for this reservoir in our reservoir data brings us 15796 records, specifically on its storage unit.

Missing and Incomplete Data.

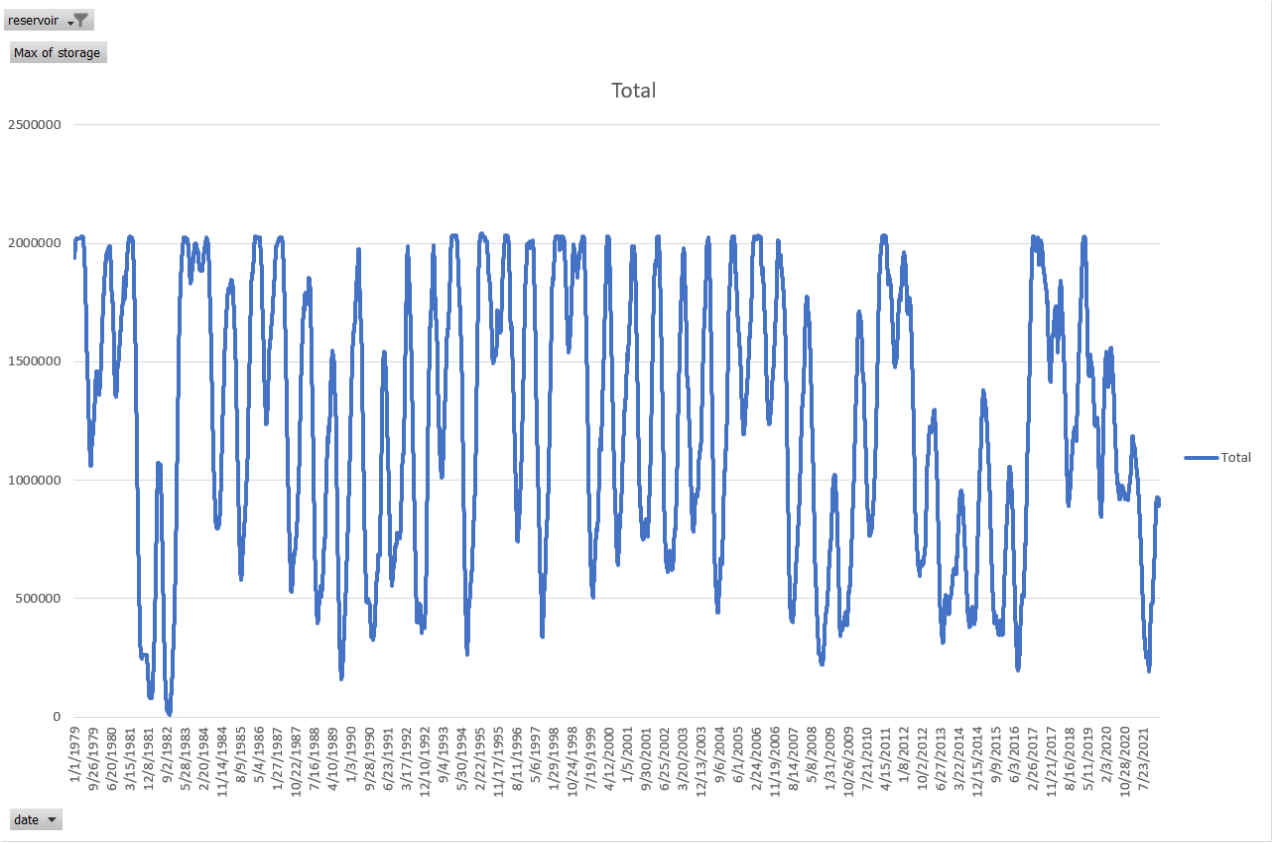
We check the storage unit by using SQL query to see if there are any data gaps between dates and it is confirmed that there are none.

Identify Outliers.

By graphing the data points shown in Figure 22, we find few outliers, notably from1981 to 1982. Upon further investigation, we discover that the outlier data happened on October 21, 1982, but our research indicates that water is released to account for the upcoming wet season. Other significant drops align with California drought.

Figure 22

Line Graph Of The San Luis Reservoir Storage Data Ranging From 1979 To 2021



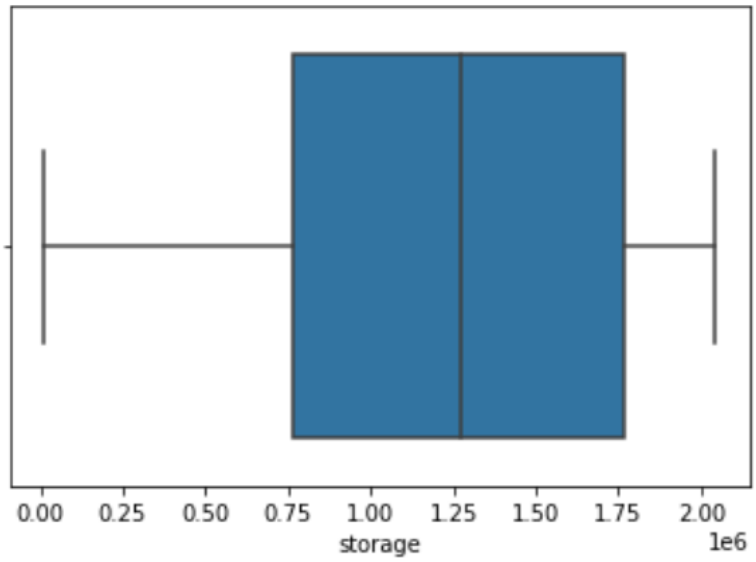
Boxplot in Figure 23 built in Python via SNS shows that there is no outlier in our reservoir_dataset

Figure 23

Box Plot Of Storage Data For San Luis Reservoir

```
sns.boxplot(x=reservoir_df['storage'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbc1cd7fb90>
```



3.4 Data Transformation

3.4.1 Data Aggregation.

After data cleaning, the snow dataset illustrates the snow data by station and date. However, for the purpose of this project, we are more concerned about the total amount of snow by date and regardless of the stations. As a result, a new snow table is created using SQLite to meet our needs. The query and new table sample are shown in Figure 24.

Figure 24

Snow Dataset After Aggregation

```
CREATE TABLE snow_data_1 as
SELECT DATE, SUM(snow_data.SWE) AS SUM_SWE FROM snow_data
GROUP BY DATE;
```

Date	SUM_SWE
Filter	Filter
2014-01-01	72.6
2014-01-02	72.7
2014-01-03	72.4
2014-01-04	72.4
2014-01-05	72.4
2014-01-06	72.5
2014-01-07	72.5
2014-01-08	72.2
2014-01-09	72.0
2014-01-10	72.0

Then, all datasets were combined by the following SQL script. The combined table contains 27 parameters: one date column, six columns that present the daily precipitation of the six chosen stations, six columns that present the daily average temperature of the six chosen stations, six columns that present the daily maximum temperature of the six chosen stations, six columns that present the daily minimum temperature of the six chosen stations, the daily sum of snow water amount at the beginning of the day of all stations, and the reservoir storage amount of San Luis reservoir. Figure 25, 26, and 27 show the samples of the first five rows in the combined table.

Figure 25

Sample Of The First Nine Columns Of The Combined Table

DATE	PRCPUSC00045118	PRCPUSC00045933	PRCPUSC00046168	PRCPUSC00049473	PRCPUSW00023233	PRCPUSW00093243	TAVGUSC00045118	TAVGUSC00045933
1/1/2014	0	0	0	0	0	0	49	57.5
1/2/2014	0	0	0	0	0	0	49	59.5
1/3/2014	0	0	0	0	0	0	49	58.5
1/4/2014	0	0	0	0	0	0	49	53.5
1/5/2014	0	0	0	0	0	0	49.5	56

Figure 26

Sample Of The Middle Nine Columns Of The Combined Table

TAVGUSC00046168	TAVGUSC00049473	TAVGUSW00023233	TAVGUSW00093243	TMAXUSC00045118	TMAXUSC00045933	TMAXUSC00046168	TMAXUSC00049473	TMAXUSW00023233
50	52.5	55	45	64	61	70	70	71
51	54	57	43.5	64	64	69	73	72
51	57	54	45	62	63	69	75	66
51.5	54	53.5	45	63	59	69	68	69
50.5	57	58	43	66	61	71	74	74

Figure 27

Sample Of The Last Nine Columns Of The Combined Table

TMAXUSW00093243	TMINUSC00045118	TMINUSC00045933	TMINUSC00046168	TMINUSC00049473	TMINUSW00023233	TMINUSW00093243	SUM_SWE	storage
63	34	54	30	35	39	27	72.6	608501
63	34	55	33	35	42	24	72.7	608922
63	36	54	33	39	42	27	72.4	609511
64	35	48	34	40	38	26	72.4	610016
63	33	51	30	40	42	23	72.4	616514

PRCP presents the daily precipitation, TAVG presents the daily average temperature, TMAX presents the daily maximum temperature, and TMIN presents the daily minimum temperature. Each of the above abbreviations is followed by a station id. Figure 28 illustrates the statistics of the combined table. We can see the maximum and minimum values of each column.

Figure 28

Statistical Summary Of The Combined Table

	count	mean	std	min	25%	50%	75%	max
PRCPUSC00045118	3012.0	2.304615e-02	0.113780	0.0	0.0	0.0	0.000	2.25
PRCPUSC00045933	3012.0	7.768426e-02	0.282418	0.0	0.0	0.0	0.000	5.50
PRCPUSC00046168	3012.0	2.834163e-02	0.139410	0.0	0.0	0.0	0.000	1.93
PRCPUSC00049473	3012.0	6.155046e-02	0.252852	0.0	0.0	0.0	0.000	3.96
PRCPUSW00023233	3012.0	2.917995e-02	0.141365	0.0	0.0	0.0	0.000	3.22
PRCPUSW00093243	3012.0	2.463811e-02	0.122821	0.0	0.0	0.0	0.000	2.47
TAVGUSC00045118	3012.0	6.433748e+01	12.568833	36.5	53.5	64.0	75.000	94.50
TAVGUSC00045933	3012.0	5.622170e+01	13.996645	25.0	45.0	55.5	68.000	91.50
TAVGUSC00046168	3012.0	6.607620e+01	12.787266	35.5	54.5	65.5	77.500	95.00
TAVGUSC00049473	3012.0	5.902332e+01	6.706668	41.0	54.5	59.5	63.500	88.50
TAVGUSW00023233	3012.0	5.961222e+01	6.315288	41.5	55.5	60.0	64.000	87.50
TAVGUSW00093243	3012.0	6.077307e+01	11.559841	33.0	51.0	60.0	71.000	90.00
TMAXUSC00045118	3012.0	7.686205e+01	15.127161	41.0	64.0	76.0	90.000	111.00
TMAXUSC00045933	3012.0	6.255063e+01	14.835672	29.0	51.0	62.0	75.000	98.00
TMAXUSC00046168	3012.0	8.227191e+01	15.854409	46.0	68.0	83.0	97.000	113.00
TMAXUSC00049473	3012.0	6.974967e+01	8.357642	50.0	64.0	69.0	74.000	110.00
TMAXUSW00023233	3012.0	6.945435e+01	8.057838	49.0	64.0	69.0	74.000	109.00
TMAXUSW00093243	3012.0	7.745169e+01	15.252516	43.0	64.0	77.0	91.000	111.00
TMINUSC00045118	3012.0	5.181292e+01	10.636673	26.0	44.0	52.0	60.000	80.00
TMINUSC00045933	3012.0	4.989276e+01	13.422736	20.0	39.0	49.0	61.000	85.00
TMINUSC00046168	3012.0	4.988048e+01	10.753827	23.0	41.0	50.0	58.000	80.00
TMINUSC00049473	3012.0	4.829698e+01	7.451309	30.0	43.0	49.0	54.000	67.00
TMINUSW00023233	3012.0	4.977009e+01	6.976975	28.0	45.0	51.0	55.000	67.00
TMINUSW00093243	3012.0	4.409446e+01	9.291905	19.0	37.0	44.0	51.000	72.00
SUM_SWE	3012.0	2.223546e+02	327.890581	0.0	0.0	74.7	345.375	1477.50
storage	3012.0	1.069579e+06	501731.690238	192050.0	653896.0	1003719.0	1467738.500	2028471.00

3.4.2 Data Normalization.

The combined table is normalized except for the Date column since the values from different columns fluctuate in different scales. We use Python in Google Colab to conduct min-max normalization so that the range of all parameters is from zero to one. Because normalization does not change the differences in the ranges of values, the features are much

easier to compare with each other. Accordingly, it can help identify the duplicated columns (if there are any). The formula applied for each column is as follows

$$x' = \frac{x - min(x)}{max(x) - min(x)} \quad (1)$$

where x' is the new value, x is the current value, min(x) is the minimum value of the column, max(x) is the maximum value of the column.

Figure 29, 30, and 31 shows the samples of the first five rows in the normalized table.

Figure 29

Sample Of The First Nine Columns In The Normalized Table

	DATE	PRCPUSC00045118	PRCPUSC00045933	PRCPUSC00046168	PRCPUSC00049473	PRCPUSW00023233	PRCPUSW00093243	TAVGUSC00045118	TAVGUSC00045933
0	1/1/2014	0.000000	0.000000	0.0	0.0	0.000000	0.000000	0.215517	0.488722
1	1/2/2014	0.000000	0.000000	0.0	0.0	0.000000	0.000000	0.215517	0.518797
2	1/3/2014	0.000000	0.000000	0.0	0.0	0.000000	0.000000	0.215517	0.503759
3	1/4/2014	0.000000	0.000000	0.0	0.0	0.000000	0.000000	0.215517	0.428571
4	1/5/2014	0.000000	0.000000	0.0	0.0	0.000000	0.000000	0.224138	0.466165

Figure 30

Sample Of The Middle Nine Columns In The Normalized Table

	TAVGUSC00046168	TAVGUSC00049473	TAVGUSW00023233	TAVGUSW00093243	TMAXUSC00045118	TMAXUSC00045933	TMAXUSC00046168	TMAXUSC00049473	TMAXUSW00023233
0	0.243697	0.242105	0.293478	0.210526	0.328571	0.463768	0.358209	0.333333	0.366667
1	0.260504	0.273684	0.336957	0.184211	0.328571	0.507246	0.343284	0.383333	0.383333
2	0.260504	0.336842	0.271739	0.210526	0.300000	0.492754	0.343284	0.416667	0.283333
3	0.268908	0.273684	0.260870	0.210526	0.314286	0.434783	0.343284	0.300000	0.333333
4	0.252101	0.336842	0.358696	0.175439	0.357143	0.463768	0.373134	0.400000	0.416667

Figure 31

Sample Of The Last Nine Columns In The Normalized Table

	TMAXUSW00093243	TMINUSC00045118	TMINUSC00045933	TMINUSC00046168	TMINUSC00049473	TMINUSW00023233	TMINUSW00093243	SUM_SWE	storage
0	0.294118	0.148148	0.523077	0.122807	0.135135	0.282051	0.150943	0.049137	0.226773
1	0.294118	0.148148	0.538462	0.175439	0.135135	0.358974	0.094340	0.049205	0.227002
2	0.294118	0.185185	0.523077	0.175439	0.243243	0.358974	0.150943	0.049002	0.227323
3	0.308824	0.166667	0.430769	0.192982	0.270270	0.256410	0.132075	0.049002	0.227598
4	0.294118	0.129630	0.476923	0.122807	0.270270	0.358974	0.075472	0.049002	0.231137

3.4.3 Data Regularization.

Next, we take the normalized combined table and apply LASSO (or L1) regularization as a feature selection step, also using Python from the Jupyter Notebook. The LASSO imposes a constraint on the parameters that might shrink the parameters' coefficients to zeros. By then, we can eliminate the corresponding parameters as a dimension reduction methodology.

We try LASSO Regularization with lambda 0.5 and 1. In both cases, all coefficients appear to be zeros, as shown in Figure 32, meaning none of the 26 parameters could be used to build a prediction model for our target.

Figure 32

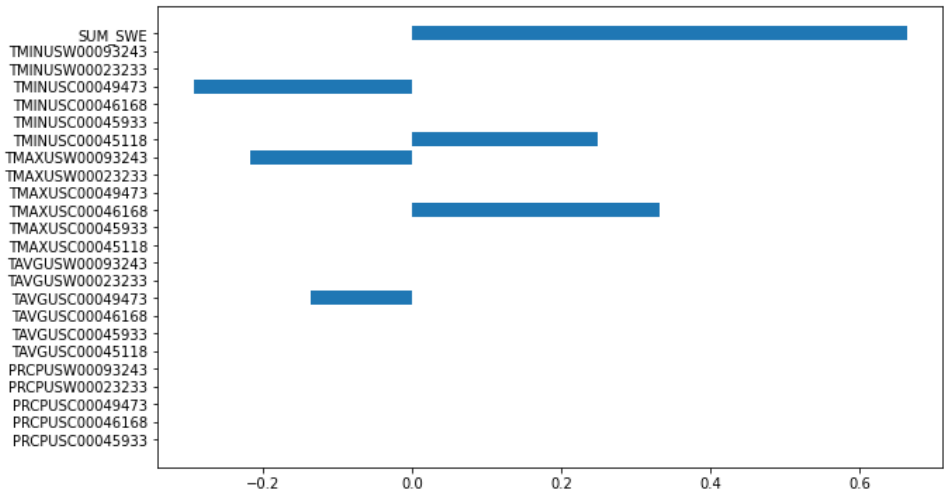
LASSO Regression Result With Lambda 0.5 and 1.

	Column_Name	Coefficient_Value
0	PRCPUSC00045933	0.0
1	PRCPUSC00046168	0.0
2	PRCPUSC00049473	0.0
3	PRCPUSW00023233	-0.0
4	PRCPUSW00093243	0.0
5	TAVGUSC00045118	-0.0
6	TAVGUSC00045933	-0.0
7	TAVGUSC00046168	-0.0
8	TAVGUSC00049473	-0.0
9	TAVGUSW00023233	-0.0
10	TAVGUSW00093243	-0.0
11	TMAXUSC00045118	-0.0
12	TMAXUSC00045933	-0.0
13	TMAXUSC00046168	-0.0
14	TMAXUSC00049473	-0.0
15	TMAXUSW00023233	-0.0
16	TMAXUSW00093243	-0.0
17	TMINUSC00045118	-0.0
18	TMINUSC00045933	-0.0
19	TMINUSC00046168	-0.0
20	TMINUSC00049473	-0.0
21	TMINUSW00023233	-0.0
22	TMINUSW00093243	-0.0
23	SUM_SWE	0.0

Replacing lambda with a very small value, such as 0.005, some coefficients began to increase (Figure 33). As a result, it might be more beneficial to keep the parameters as they are to process the next step: Data Preparation.

Figure 33

LASSO Regression Result With Lambda 0.005.



3.5 Data Preparation

After establishing the transformed dataset we began data preparation. The first step is to establish what our independent variables are and what our prediction targets are. We have about 26 independent variables in our combined_dataset consisting of about 3012 rows worth of data. With the storage attribute from our reservoir data being our prediction target. The attributes consist of precipitation and snowmelt from multiple sensor stations around a 50 mile radius from San Luis Reservoir. Looking at the dataset we decided to use the simple 80/20 strategy. This is 80% training and 20% testing data subsets, and KFold Cross Validation split the data into validation sets later in the model development phase. For preparing the data we split the data into two different data subsets consisting of training and testing data subsets. Figure 34 shows the shape of the input and target training datasets, input and target validation datasets, and input and target testing datasets respectively.

Figure 34

The Shapes Of The Training, Validation, and Testing data

```
print(X_train.shape), print(y_train.shape)
print(X_valid.shape), print(y_valid.shape)
print(X_test.shape), print(y_test.shape)
```

```
(1446, 26)
(1446,)
(963, 26)
(963,)
(603, 26)
(603,)
(None, None)
```

In order to ensure that the training of the machine learning model is robust enough to satisfy a satisfactory machine learning model we utilized 60% of our data as training data. This consists of 1446 data rows for our training. A sample of it is shown in Figure 35.

Figure 35

The First And Last Five Rows In The Training Dataset

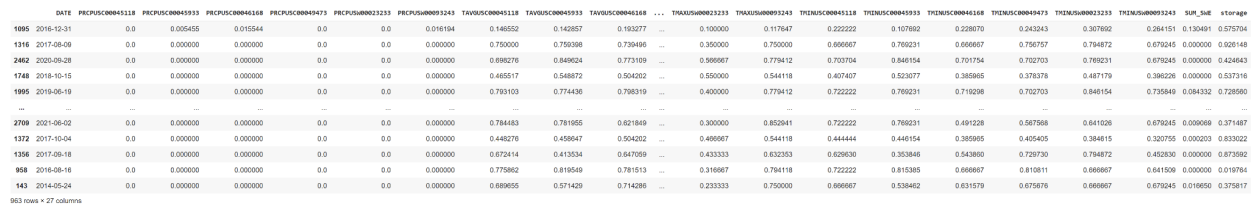
	DATE	PRCPUSC00045118	PRCPUSC00045933	PRCPUSC00046168	PRCPUSC00049473	PRCPUSW00023233	PRCPUSW00093243	TAVGUSC00045118	TAVGUSC00045933	TAVGUSC00046168	...	TMAXUSW00023233	TMAXUSW00093243	TMINUSC00045118	TMINUSC00045933	TMINUSC00046168	TMINUSC00049473	TMINUSW00023233	TMINUSW00093243	SUM_SWE	storage
2551	2020-12-26	0.0	0.034545	0.000000	0.040404	0.006211	0.000000	0.051724	0.210526	0.176471	...	0.200000	0.294118	0.166667	0.246154	0.210526	0.351351	0.461538	0.377358	0.114585	0.407462
842	2016-04-22	0.0	0.065455	0.000000	0.000000	0.068323	0.068826	0.525862	0.233083	0.537815	...	0.283333	0.367647	0.518519	0.261538	0.543860	0.783784	0.641026	0.547170	0.292927	0.438904
760	2016-01-31	0.0	0.000000	0.243523	0.055556	0.121118	0.214575	0.241379	0.075188	0.260504	...	0.100000	0.176471	0.314815	0.107692	0.385965	0.486486	0.435897	0.452830	0.401760	0.272798
2477	2020-10-13	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.551724	0.714286	0.613445	...	0.616667	0.676471	0.500000	0.707692	0.456140	0.567568	0.589744	0.490566	0.000000	0.417479
684	2015-11-16	0.0	0.000000	0.000000	0.055556	0.000000	0.000000	0.293103	0.135338	0.302521	...	0.200000	0.250000	0.351852	0.138462	0.403509	0.324324	0.384615	0.358491	0.046159	0.111421
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2257	2020-03-07	0.0	0.009091	0.000000	0.025253	0.003106	0.008097	0.344828	0.157895	0.327731	...	0.166667	0.279412	0.333333	0.200000	0.263158	0.486486	0.538462	0.471698	0.228291	0.653292
2957	2022-02-05	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.241379	0.345865	0.277311	...	0.366667	0.294118	0.185185	0.369231	0.263158	0.162162	0.256410	0.226415	0.342538	0.391740
2394	2020-07-22	0.0	0.000000	0.000000	0.000000	0.006211	0.000000	0.663793	0.661654	0.714286	...	0.383333	0.735294	0.592593	0.676923	0.614035	0.702703	0.794872	0.603774	0.000000	0.434607
1409	2017-11-10	0.0	0.000000	0.000000	0.002525	0.000000	0.000000	0.431034	0.315789	0.495798	...	0.316667	0.411765	0.518519	0.338462	0.473684	0.567568	0.538462	0.490566	0.012453	0.677130
959	2016-08-17	0.0	0.000000	0.000000	0.002525	0.000000	0.000000	0.810345	0.842105	0.840336	...	0.350000	0.882353	0.759259	0.861538	0.736842	0.810811	0.666667	0.679245	0.000000	0.021566

1446 rows × 27 columns

Then split off 20% of our data into validation data so that we can test accuracy of the model before we begin testing the data. The validation dataset consists of 963 data rows for validation. A sample of it is shown in Figure 36.

Figure 36

The First And Last Five Rows In The Validation Dataset



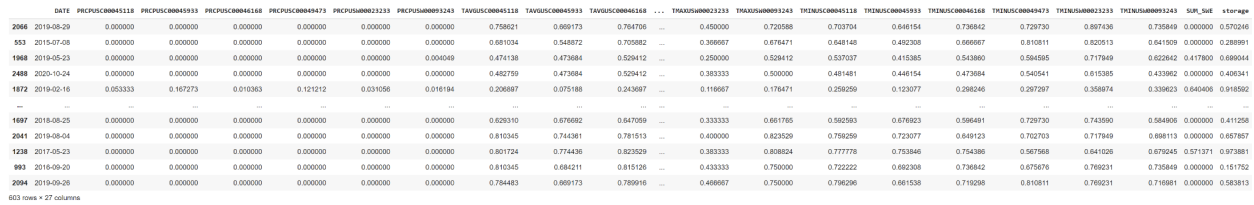
	DATE	PRCPUSC00045118	PRCPUSC00045933	PRCPUSC00046168	PRCPUSC00049473	PRCPUSW00023233	PRCPUSW00093243	TAVGUSC00045118	TAVGUSC00045933	TAVGUSC00046168	...	TMAXUSW00023233	TMAXUSW00093243	TMINUSC00045118	TMINUSC00045933	TMINUSC00046168	TMINUSC00049473	TMINUSW00023233	TMINUSW00093243	SUM_SWE	storage
1095	2016-12-31	0.0	0.005455	0.015544	0.0	0.0	0.016194	0.146552	0.142857	0.193277	...	0.100000	0.117647	0.222222	0.107692	0.228070	0.243243	0.307692	0.264151	0.130491	0.575704
1316	2017-08-09	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.750000	0.759398	0.739496	...	0.350000	0.750000	0.666667	0.769231	0.666667	0.756757	0.794872	0.679245	0.000000	0.926148
2462	2020-09-28	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.698276	0.849624	0.773109	...	0.566667	0.779412	0.703704	0.846154	0.701754	0.702703	0.769231	0.679245	0.000000	0.424643
1748	2018-10-15	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.465517	0.548872	0.504202	...	0.550000	0.544118	0.407407	0.523077	0.385965	0.378378	0.487179	0.396226	0.000000	0.537316
1995	2019-06-19	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.793103	0.774436	0.798319	...	0.400000	0.779412	0.722222	0.769231	0.719298	0.702703	0.846154	0.735849	0.084332	0.728560
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2709	2021-06-02	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.784483	0.781955	0.621849	...	0.300000	0.852941	0.722222	0.769231	0.491228	0.567568	0.641026	0.679245	0.099069	0.371487
1372	2017-10-04	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.448276	0.458647	0.504202	...	0.466667	0.544118	0.444444	0.446154	0.385965	0.405405	0.384615	0.320755	0.000203	0.833022
1356	2017-09-18	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.672414	0.413534	0.647059	...	0.433333	0.632353	0.629630	0.353846	0.543860	0.729730	0.794872	0.452830	0.000000	0.873592
958	2016-08-16	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.775862	0.819549	0.781513	...	0.316667	0.794118	0.722222	0.815385	0.666667	0.810811	0.666667	0.641509	0.000000	0.019764
143	2014-05-24	0.0	0.000000	0.000000	0.0	0.0	0.000000	0.689655	0.571429	0.714286	...	0.233333	0.750000	0.666667	0.538462	0.631579	0.675676	0.666667	0.679245	0.016650	0.375817

963 rows × 27 columns

The last 20% of the data is testing data which does not be touched until our model is ready for testing. This consists of 963 data rows for testing data. A sample of it is shown in Figure 37.

Figure 37

The First And Last Five Rows In The Testing Dataset



	DATE	PRCPUSC00045118	PRCPUSC00045933	PRCPUSC00046168	PRCPUSC00049473	PRCPUSW00023233	PRCPUSW00093243	TAVGUSC00045118	TAVGUSC00045933	TAVGUSC00046168	...	TMAXUSW00023233	TMAXUSW00093243	TMINUSC00045118	TMINUSC00045933	TMINUSC00046168	TMINUSC00049473	TMINUSW00023233	TMINUSW00093243	SUM_SWE	storage
2066	2019-08-29	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.758621	0.669173	0.764706	...	0.450000	0.720588	0.703704	0.646154	0.736842	0.729730	0.897436	0.735849	0.000000	0.570246
553	2015-07-08	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.681034	0.548872	0.705882	...	0.366667	0.676471	0.648148	0.492308	0.666667	0.810811	0.820513	0.641509	0.000000	0.288991
1968	2019-05-23	0.000000	0.000000	0.000000	0.000000	0.000000	0.004049	0.474138	0.473684	0.529412	...	0.250000	0.529412	0.537037	0.415385	0.543860	0.594595	0.717949	0.622642	0.417800	0.699044
2488	2020-10-24	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.482759	0.473684	0.529412	...	0.383333	0.500000	0.481481	0.446154	0.473684	0.540541	0.615385	0.433962	0.000000	0.406341
1872	2019-02-16	0.053333	0.167273	0.010363	0.121212	0.031056	0.016194	0.206897	0.075188	0.243697	...	0.116667	0.176471	0.259259	0.123077	0.298246	0.297297	0.358974	0.339623	0.640406	0.918592
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1697	2018-08-25	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.629310	0.676692	0.647059	...	0.333333	0.661765	0.592593	0.676923	0.596491	0.729730	0.743590	0.584906	0.000000	0.411258
2041	2019-08-04	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.810345	0.744361	0.781513	...	0.400000	0.823529	0.759259	0.723077	0.649123	0.702703	0.717949	0.698113	0.000000	0.657857
1238	2017-05-23	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.801724	0.774436	0.823529	...	0.383333	0.808824	0.777778	0.753846	0.754386	0.567568	0.641026	0.679245	0.571371	0.973881
993	2016-09-20	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.810345	0.684211	0.815126	...	0.433333	0.750000	0.722222	0.692308	0.736842	0.675676	0.769231	0.735849	0.000000	0.151752
2094	2019-09-26	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.784483	0.669173	0.789916	...	0.466667	0.750000	0.796296	0.661538	0.719298	0.810811	0.769231	0.716981	0.000000	0.583813

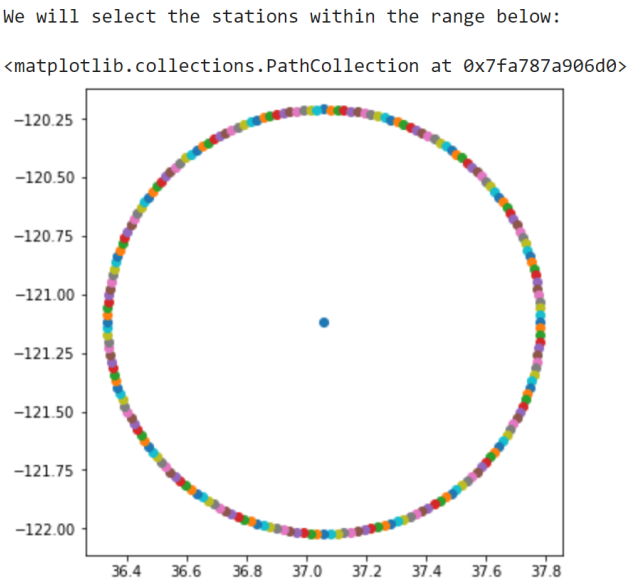
603 rows × 27 columns

3.6 Data Statistics

Through our work with weather stations, we grab a total of 1808 weather stations, but not all of the weather stations are within the scope of our project and some do not include the data points we want. After 50 mile buffer, we reduce it down to 95 stations bringing about 170,000 records. Figure 38 shows the coordinates of where the 50 mile buffer covers.

Figure 38

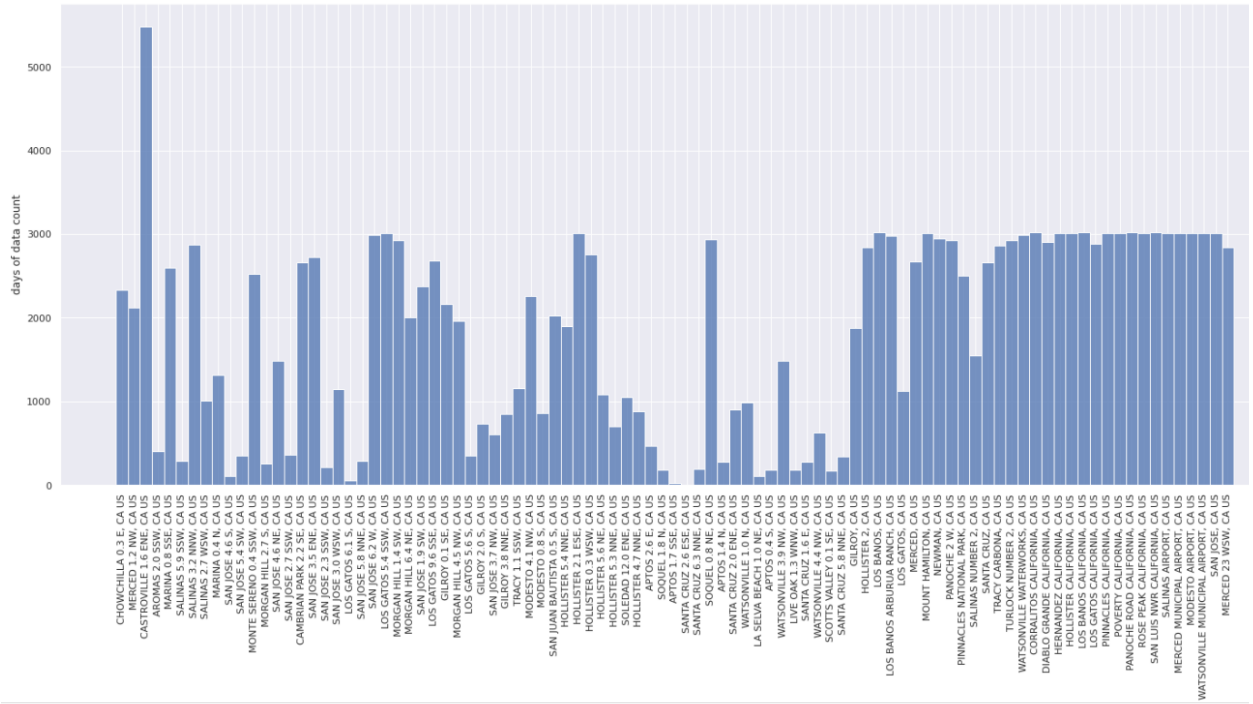
Latitude And Longitude Range For The Station Selection



Through our data range, there are 3016 days and our expectation that each station would have 3016 records, but through our research we find that some stations have missing records and one has more records than it should be, as shown in Figure 39. To address this we further limit stations with those who have a 90% completed rate, giving us about 2714 days of record per station.

Figure 39

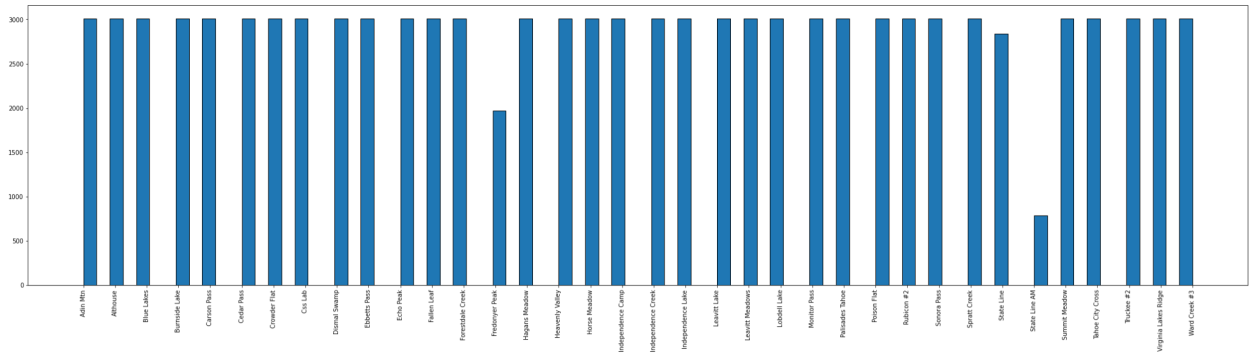
Stations And Date Frequency



In our next parameter, Snow Data, we also limit it to the same dates but unlike weather almost every station has a 100% completion rate. There are outlier stations that do have a sizable amount of outliers but they are removed as stated in data pre-processing. In addition, Fredonyer Peak is removed as it only has data from 2016 going forward which do not meet our 90% threshold so it is also removed. Figure 40 shows the reporting frequency of the snow stations.

Figure 40

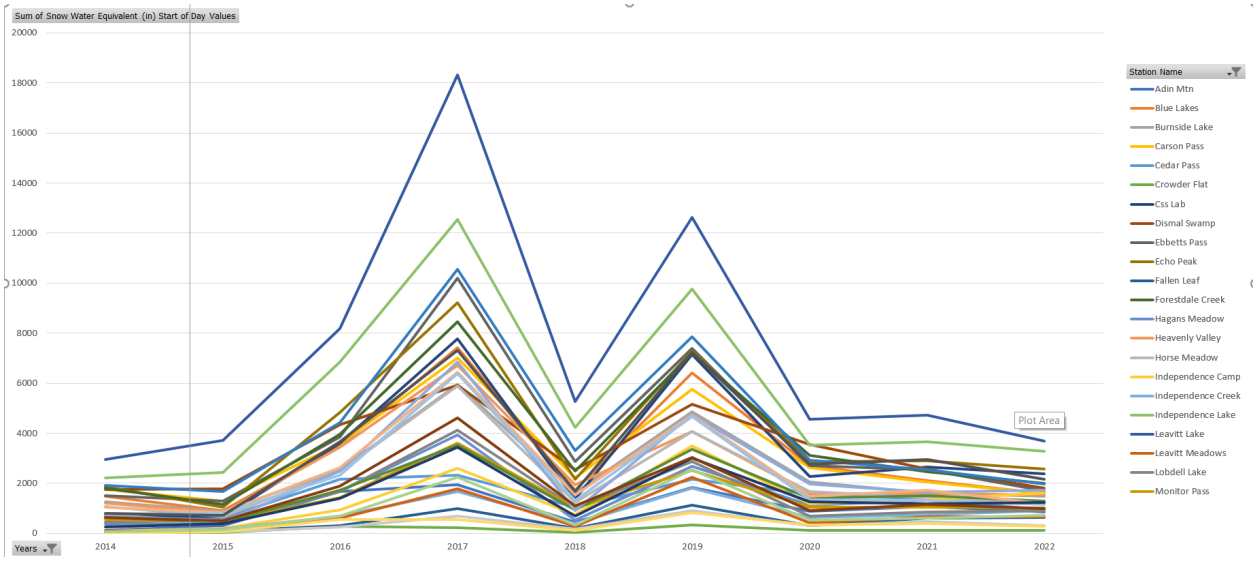
Stations And Reporting Frequency



After removing all the unnecessary data in pre-processing, we then graph all of the records in relation to their stations. We want to make sure that the rate of change is within the scope of our understanding on how California's climate works and there are no outliers we miss. To identify outliers we use a line graph shown in Figure 41 to confirm this.

Figure 41

Graph Representation Of Each Station.

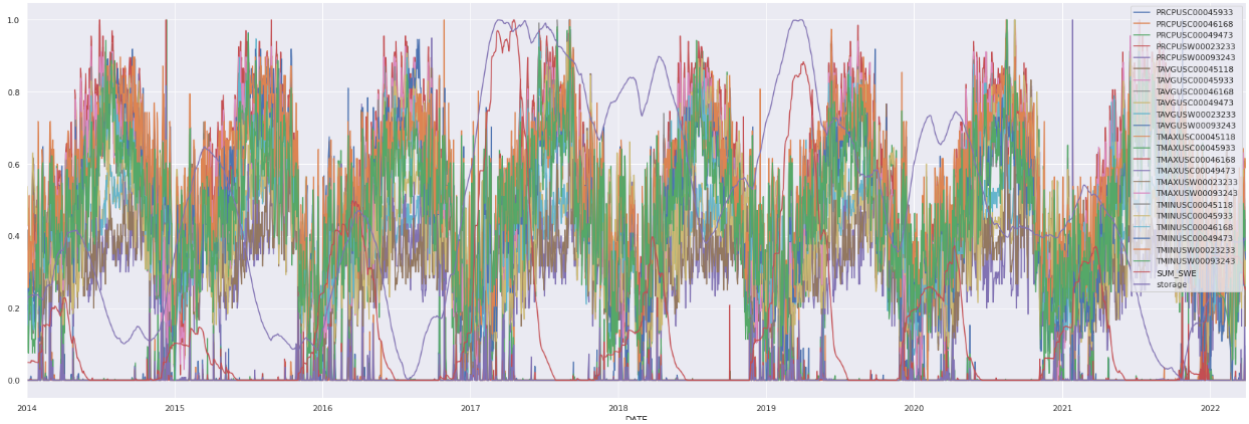


Santa Clara County has all of their imported water, from the Sierra snowpack, go through the San Luis reservoir. This allows us to limit our data to just the San Luis Reservoir on the dataset we import from the (CNRA). With the focus of our project on the San Luis Reservoir we are able to remove all other reservoirs from our reservoir dataset which reduce the data from 856995 to 15796. We are able to further reduce the dataset down to storage and dates as all of the other attributes do not contribute any beneficial information to our data. Within the data subset we have no null values, but we do have outliers as stated in pre-processing. The comparison of the outlier dates and real world events allow us to verify and accept the outliers as part of the data as it matches California droughts and wet season periods.

We combine all cleaned dataset into a combined table and conduct data normalization. Figure 42 shows the distribution of the 26 parameters in the normalized table from January 1, 2014 to March 31, 2022. Observe that all of the parameters scale to the range of zero to one, which makes it much easier to see the patterns and to compare the trends with each other.

Figure 42

Line Chart for 26 Parameters in The Normalized Table from January 1, 2014 to March 31, 2022.

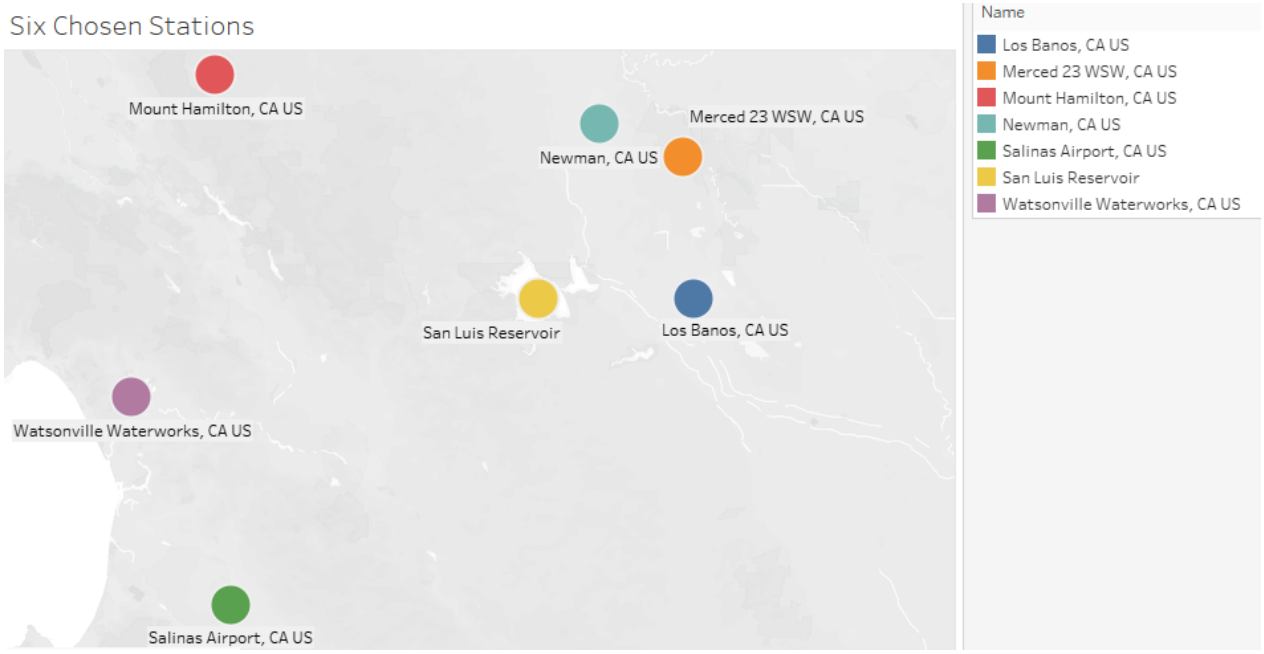


3.7 Data Analytics Results

The six stations that we use as parameters for precipitation and temperature are shown in map in Figure 43. Those stations are within 50 miles of the San Luis Reservoir and have sufficient data for our project.

Figure 43

Six Chosen Stations



Apply linear regression to the normalized table to test the assumption in the data transformation section that linear regression might not work well for our data. The R^2 of the linear regression line shown in Figure 44 is about -1.088 which is smaller than zero and it consolidates the assumption. Results presented in the bar chart shown in Figure 45 further proves that linear regression does not fit well with our data.

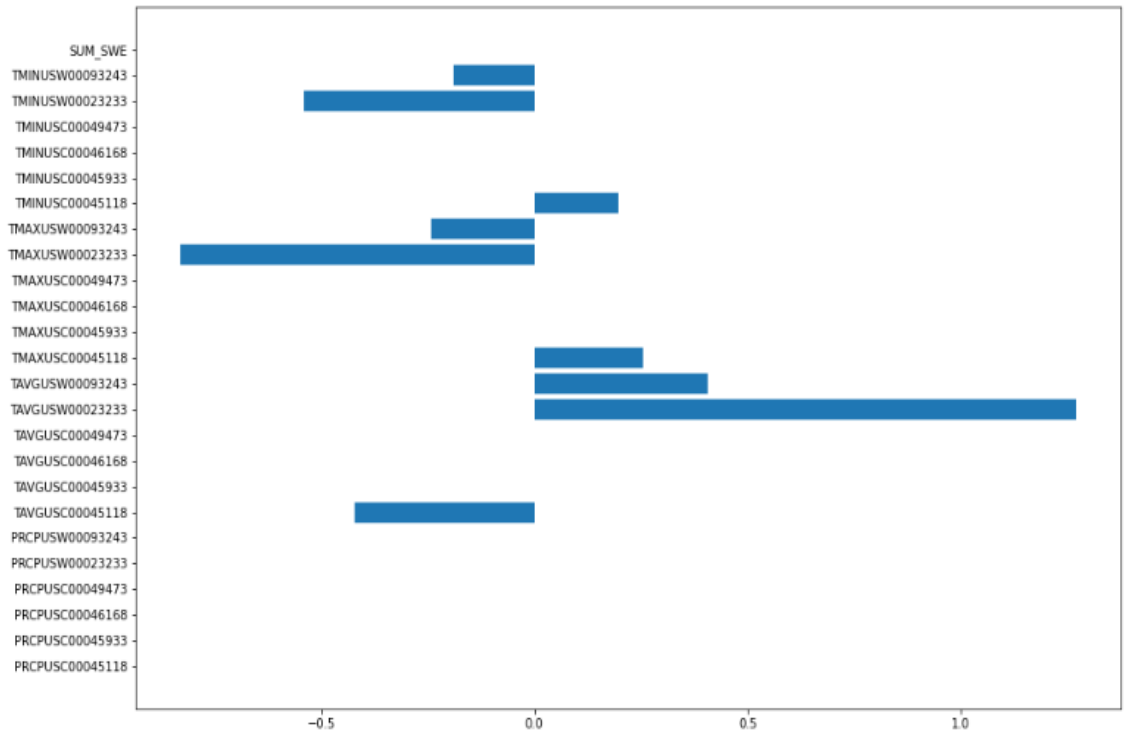
Figure 44

Result of Applying Linear Regression On The Normalized Table

```
R squared Error on test set:  -1.087750594962471
        Column_Name  Coefficient_Value
0   PRCPUSC00045118      -1.847189e-01
1   PRCPUSC00045933       3.481366e-02
2   PRCPUSC00046168       8.138944e-02
3   PRCPUSC00049473      -1.486539e-02
4   PRCPUSW00023233      -3.255815e-01
5   PRCPUSW00093243       2.493298e-01
6   TAVGUSC00045118      -4.206021e+08
7   TAVGUSC00045933       1.209487e+03
8   TAVGUSC00046168       9.799515e+01
9   TAVGUSC00049473       4.154358e+02
10  TAVGUSW00023233       1.270945e+09
11  TAVGUSW00093243       4.070166e+08
12  TMAXUSC00045118       2.538116e+08
13  TMAXUSC00045933      -6.272629e+02
14  TMAXUSC00046168      -5.448997e+01
```

Figure 45

Result of Applying Linear Regression On The Normalized Table In Bar Chart



4. Model Development

4.1 Model Proposal

In this section, proposals for XGBoost, Gradient Boost, Random Forest, and Support Vector Regression (SVR) are presented. XGBoost, Gradient Boost, and Random Forests models share the same input data, training and testing data split, and evaluation metrics. Therefore, they are discussed together. The SVR model is developed using different input data and is discussed separately.

For XGBoost, Gradient Boost, Random Forest Model.

The target problem for these three machine learning models is to predict water storage level for San Luis Reservoir. The dataset that XGBoost, Gradient Boost, and Random Forest utilized is the combined table that is preprocessed in the Data Engineering phase. The combined table contains 25 attributes including daily snow water equivalent amount for the Sierra area, daily precipitation, temperature minimum, temperature maximum, and temperature average of the six stations within the 50 miles of the reservoir. After the variables are determined, the dataset is then split into 80% training dataset and 20% testing dataset.Permutation feature importance is performed as well to inspect the models.

In order to enhance predictions and to minimize errors between the predicted results and the actual results, grid searches are performed for each scenario to tune its hyperparameters so that the predicted results can be more accurate. Cross validations are performed after the grid search to prevent the model from failing to predict any unseen data. For all models, we would like to have the predicted results be as close as possible to the actual target values and follow the overall trend of the actual target values. There are many different metrics used to measure how well the model performs. Four metrics that are popular in the machine learning field are

coefficient of determination score (R^2), root mean square error ($RMSE$), mean square error (MSE), mean absolute error (MAE), and mean error (ME) in percentage. Their purpose is to assess how well each scenario fits its dataset. The metrics are defined in the Model Evaluation Method section. Predicted results are also plotted out for each scenario to provide a visual estimate on how well the model performs or in other words how well the predictions are. Specifically for the SVR model, the dataset for each scenario is split into training and testing datasets every six months to see how the model reacts to different sizes of training dataset. Table 8 highlights the differences between each of the algorithms utilized. It also highlights how each model was trained, tested and validated. It also shows the evaluation metrics and how we tuned each of the hyperparameters for each of the algorithms.

Table 8

Overview Comparisons of All Four Models' Proposal

Machine Learning Algorithm	XG Boosting	Gradient Boost	Random Forest	SVR
Number of Input Variables	25	25	25	2 or 4
Training and Testing Ratio	80:20	80:20	80:20	Different Ways
Tuning Hyperparameters Method	Grid Search	Grid Search	Grid Search	Grid Search
Evaluation Metrics	R^2, RMSE and MSE	R^2, RMSE and MSE	R^2, RMSE and MSE	R^2, RMSE, MAE, ME
Validation Method	10-Fold Cross Validation for All Models			

For SVR Model.

There are two scenarios developed for this model. For each scenario, instead of using various associated features of San Luis Reservoir such as weather, precipitation, the model utilizes the historical water storage level as the main feature and snow water equivalent amount

in the Sierra area as the secondary feature. The following paragraph describes the details for each scenario.

The first scenario only considers the historical water storage level of the reservoir. The historical data is aggregated to the average of the water storage level for the past seven days and the average of the water storage level for the past 30 days. The second scenario considers both the historical water storage level of the reservoir and the sum of the snow water equivalent amount in the Sierra area. The two features are aggregated to four different variables, two for each feature. The four variables are the sum of the snow water equivalent amount for the Sierra area for the past seven days, the sum of the snow water equivalent amount for the Sierra area for the past 30 days, the average of the water storage level for the past seven days, and the average of the water storage level for the past 30 days. After the input variables are set for each scenario, each scenario goes through splitting the data. Because the model factors in time and uses the recent historical data to predict the near future, when splitting data for training and testing datasets, the data cannot be split randomly.

The data we have covers from the beginning of 2014 to the end of March of 2022 (99 months of data in total) and is initially split to testing and training dataset to just predict the end of 15 months, meaning data before January 1st, 2021 is the training dataset and data starting from January 1st 2021 is the testing dataset. After both scenarios are done splitting its own data into training and testing datasets for an initial prediction, the machine employs SVR, and goes through its learning process using data in the training dataset, then performs predictions using input data in the testing dataset and compares its predicted results to the target values corresponding to the input data.

4.2 Model Supports

Description of the Platform, Environment, and Tools.

The models are built on Google Colab. Google Colab is an interactive computing platform that works very similar to Jupyter and using it will not incur a fee. The reason why Google Colab is chosen over Jupyter is because Google Colab runs entirely in the cloud with no setups required and on the other hand, Jupyter requires different setups on local computers. Running programs on Google Colab does not occupy local random access memory (RAM) and reduces the risk of data loss through automatic saves. Furthermore, Google Colab does not require management of the Python environments and libraries. This increases efficiency as it uses less storage space on local machines and less internet bandwidth to download Python packages.

The platform and environment described above are important, so are all the Python library packages used. Using Python library packages effectively not only saves tremendous time on programming, but also decreases the chance of having calculation mistakes because sometimes the math can be complicated and lengthy. The following paragraph describes the library packages used for this model.

In order to access all the data in Google Drive through Google Colab we required access authorization through the google.colab.drive library. Pandas, Numpy, and Funtools help on data manipulation, data transformation, and data calculation, etc. Scikit-Learn is another very useful library that helps establish machine learning models. Matplotlib and Seaborn are used for data visualizations.The Python library packages used for this project are shown in Table 9.

Table 9

Python Libraries Used for Models

Library	Specific Methods	Usage
Google	.colab.drive	Mounting Google Drive to GC platform
Pandas	n/a	Data and data frame manipulations
Numpy	n/a	Data calculations and manipulations
Functools	.reduce	Data calculations
Scikit-Learn	.model_selection.train_test_split	Splitting data
	.model_selection.cross_val_score	Cross validation
	.model_selection.GridSearchCV	Grid search the best hyperparameters
	.svm.SVR	Building support vector regression model
	.ensemble.RandomForestRegressor	Building support vector regression model
	.ensemble.GradientBoostingClassifier	Building support vector regression model
	.preprocessing.StandardScaler	Standardizing data
	.model_selection.RepeatedKFold	Repeating the k-fold function
	.metrics.mean_absolute_error	Calculating mean absolute error
	.inspection.permutation_importance	Determining Feature Importance
Xgboost	n/a	For XGBoosting Modeling
Matplotlib	n/a	Data visualizations
Seaborn	n/a	Data visualizations

Description of Model Architecture, Components, and Dataflows.

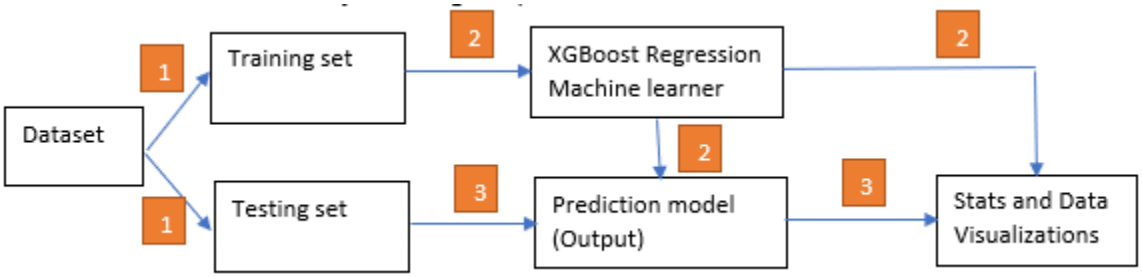
XGBoosting Model.

Figure 46 shows the dataflow for the XGBoosting model. Step one in the data flow diagram shows this process. In step two, the training set is used to train a XGBoost Regression prediction model with the hyperparameters setup that can avoid overfitting. In addition, some evaluation metrics such as R^2 are calculated on the prediction model and the training set to see

how the new model fits the training dataset. Step three shows that the testing is employed to test the prediction model. Some evaluations, statistics, and data visualizations are returned as the results of the testing process. A note on the reason for not having a validation dataset for the Model Development phase: validation sets will be split inside the training dataset when conducting K-fold Cross Validation. The data flows for the Model Development phase are shown below.

Figure 46

Architecture and Data Flows Diagram.



Gradient Boosting Model.

Figure 47 shows the dataflow for Gradient Boosting Model. Normalized dataset is first imputed into the dataflow and refined by removing unnecessary columns. The product of the refinement is stored in a temporary memory, on Google Colab, in the "Dataset used for Machine Learning Model" section. Leveraging the convenience of sklearns train_test_split Python library, we split the dataset into a 80:20 split with the testing dataset being the 20%. As this is intended to do a regression of the input parameters to the reservoir storage level, shuffle is turned on for train_test_split to remove any possible temporal attribute.

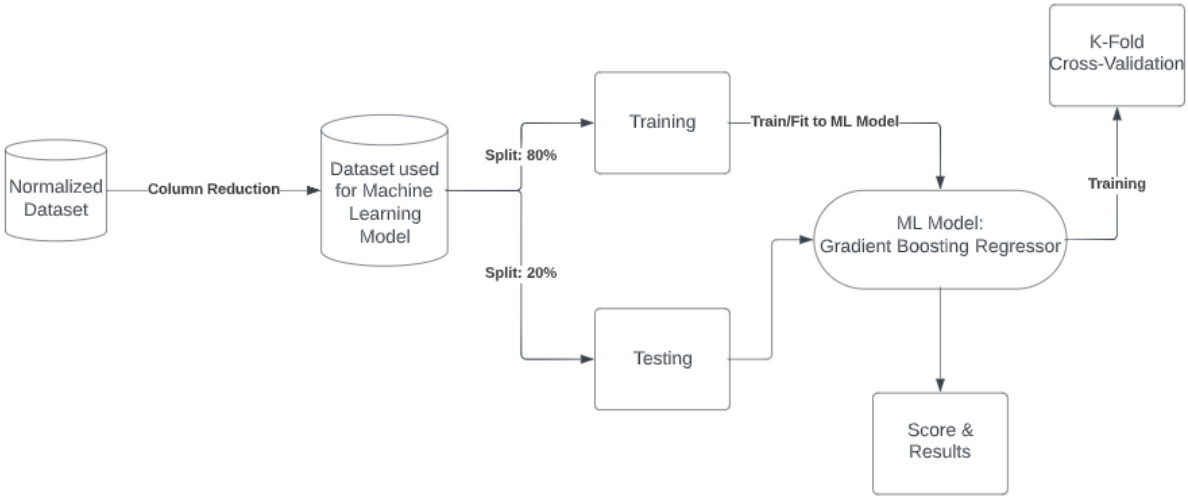
Though there is a spatial aspect to California's water supply chain, we choose to ignore that as we try to find a relationship between precipitation/snowfall with reservoir level. Validation will be done through K-fold cross-validation utilizing Scikit-Learns Python library

thus separating the full dataset into Validation and Test is unnecessary. The reason it is unnecessary is because K-fold splits the overarching Train dataset into Train and Validation during cross-validation leaving Test data untouched for evaluation purposes.

This paragraph explains the model architecture diagram shown in Figure 47. Training, 80%, is run through the machine learning model for training and fitting purposes to output predictions. This dataset, in conjunction with the machine learning model, is used in the K-Fold cross-validation. Testing, 20%, is only used on the machine learning model to output predictions.

Figure 47

Diagram of Machine Learning Architecture



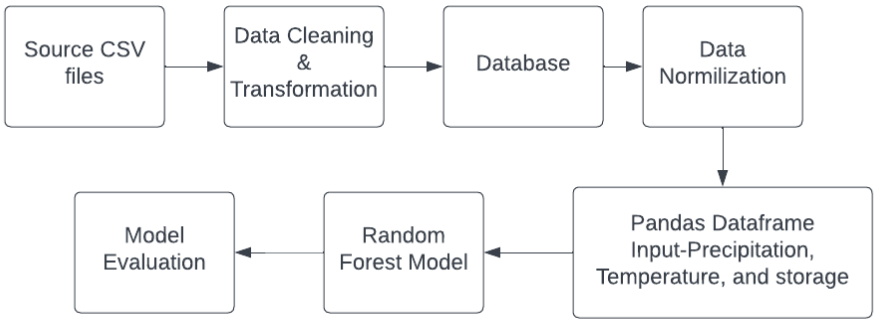
Random Forest Model.

Figure 48 shows a data flow diagram for the random forest model. The data was retrieved from the normalized dataset and was pulled into a Pandas dataframe. From there the data was split into an 80:20 split utilizing the train test split function within Scikit Learn. Validation will occur on the training dataset before the training is used on the actual model for training. After validation has been completed the data will then go into the Random Forest model where it will

be used to train the model and subsequently test the model. Using the evaluation methods will be the last stage to determine the model accuracy.

Figure 48

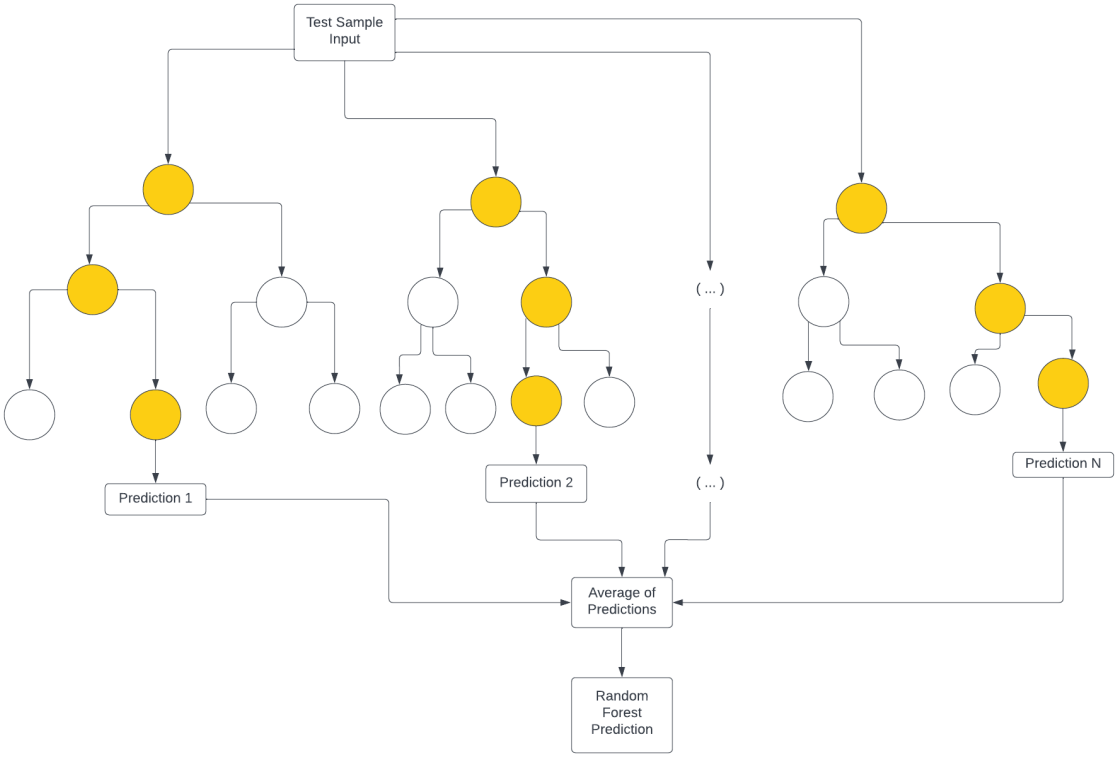
The Data Flow Diagram of Random Forest



Conceptually, Random Forest is one of the most useful algorithms for this purpose due to its versatility to work with both large datasets and small datasets and its ability to minimize the chances of overfitting the data. The algorithm leverages several different instances of decision trees to produce the output results which is an average of the outputs from all of the decision trees. This is what is referred to as a bagging technique which trains a model using several different models in order to increase accuracy. How the Random Forest algorithm utilizes multiple trees is by randomly selecting different features for each tree and splitting the tree up based on feature importance within that specific tree. Then choosing the best path within each tree and averaging up all of the solutions provided by the trees within the forest results in a random forest prediction. A conceptual diagram of a random forest algorithm is shown below in Figure 49. Where each highlighted node represents the path that the algorithm took through each of the trees. Each of the node splits are based upon feature importance evaluations as well as sample sizes per each feature.

Figure 49

A Conceptual Diagram of Random Forest Algorithm



SVR Model.

SVR was an extended form of SVM which was initially developed by Vapnik and his colleagues to solve binary classification problems (Smola & Schölkopf, 2004; Zhang & O'Donnell, 2020). As its popularity grew, SVR was later developed, also by Vapnik and his colleagues, to solve regression problems (Zhang & O'Donnell, 2020). Therefore, to better understand SVR, it is more appropriate to start with understanding SVM.

The main idea of SVM is to map the transformed nonlinear input data to a high dimensional feature space, also known as the kernel space, and to look for a linear high

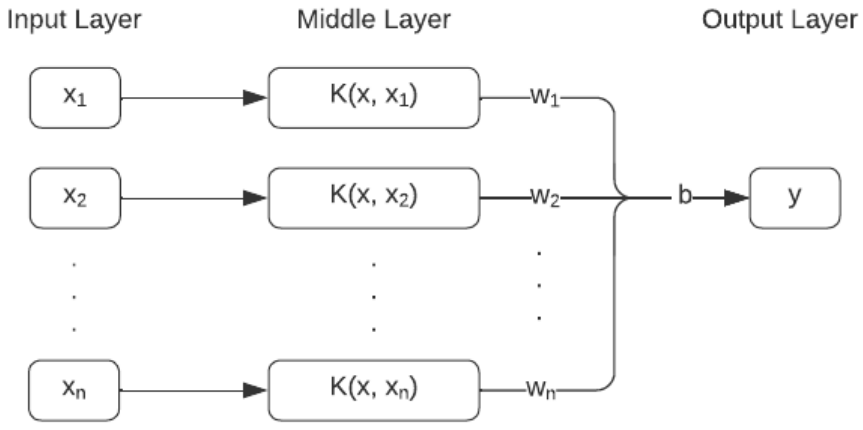
dimensional plane (hyperplane) within the space to separate the observed points (Vapnik, 1995). The hyperplane is known as the kernel plane and this process is known as the kernel trick. Figure 50 shows an overview of the architecture of SVM adapted from Smola and Schölkopf (2004), where the input layer is the input variables, the middle layer is where the kernel function is applied to each variable in the input layer so that the linear relationship can be found in a high dimensional space. The result of the linear relationship found is a hyperplane. The w_i is the weight corresponding to each of the transformed vectors and connecting to the output layer, b is the bias which yields the target variable in the output layer. Therefore, the hyperplane y can be represented as

$$y = \sum_{i=1}^{n} w_i K(x, x_i) + b \tag{2}$$

where w vector controls the slope of the function.

Figure 50

Architecture of Support Vector Machine adapted from Smola and Scholkopf (2004)



However, for regression problems, ε-insensitive loss function was introduced to approximate the hyperplane y so that all the predicted points would still be within the preset ε range (Zhang & O'Donnell, 2020). The ε range is often referred as the ε-insensitive tube and we would like to have the tube as flat as possible, meaning to obtain a smaller slope, in other words, to obtain a smaller w vector. Hence, the problem can be considered as to minimize the length of vector w as follows:

$$min\ \frac{1}{2}||w||^2 \text{ constrained by } wx_i + b - \varepsilon \leq y_i \leq wx_i + b + \varepsilon \tag{3}$$

The w minimization cannot be done without the kernel trick and to make the kernel trick happen, we need a kernel function to trigger the transformation. Many kernel functions are developed already in the field of study and there are three kernel functions borrowed for this model, which are (a) linear kernel function, defined as

$$K(x, x_i) = (x \cdot x_i) + c \tag{4}$$

where $(x \cdot x_i)$ represents the dot product between feature vector x and x_i, and c is a constant; (b) polynomial kernel function defined as

$$K(x_i, x_j) = (x \cdot x_i)^p \tag{5}$$

where $(x \cdot x_i)$ represents the dot product between the feature vector x and x_i, and p is the degree of the polynomial; and (c) radial basis function defined as

$$K(x, x_i) = e^{-\gamma||x - x_i||^2} \tag{6}$$

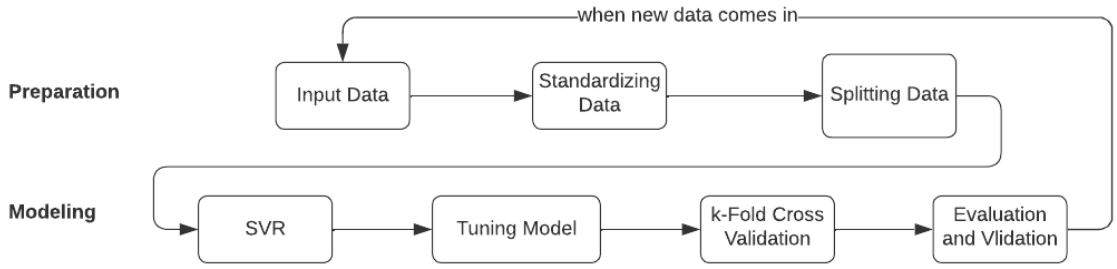
where $||x - x_i||^2$ represents the squared distance between feature vector x and x_i, γ is a parameter defined as $\frac{1}{2\sigma^2}$ where is σ a free parameter. Depending on which kernel function is

applied, the hyperparameters differ. For all three kernel functions mentioned, they all have a regularization parameter C, which has to be bigger than zedo, and parameter ε that indicates the insensitivity hinge of the loss function. The higher the C level is, the harder the marin it is, meaning there is less room for errors. The smaller the ε is, the more support vectors there would be or in other words the more tolerant the model is and vice versa. Other than the regularization and insensitivity hinge loss parameter, polynomial, and radial basis kernel have another parameter need to tune to optimize the prediction which is p, and γ respectively.

For both scenarios, the input variables mentioned in the Data Proposals section, first goes through standardization so that each variable has values with mean of zero and standard deviation of one. After standardizing the data, the data is split into training and testing dataset. Then an SVR model is generated using Scikit-Learn pre-built method and the splitted data is fitted to the model going through the layers of transformation shown in Figure 50 and trying the three different kernel functions to find the best one that fits its dataset. Grid search is performed after fitting the model to find the best hyperparameters for the model. Cross validation is also performed for the tuned model with the best fitted hyperparameters to make sure all data can be handled by the model. Finally, the output predictions are evaluated by different metrics. Figure 51 shows an overview of how the dataflow looks like for the model.

Figure 51

Dataflow of Model for Both Scenarios



4.3 Model Comparisons and Justifications

In this section, the comparisons between our models and other models in the field of study are made. Justifications are also made for each model.

XGBoosting Model.

XGBoost's main target is to optimize a regularization objective function that contains a combination of a convex loss function and a penalty term of the complexity of the model (Chen & Guestrin, 2016). The regularization term also helps prevent overfitting. Equation 7, adopted from Chen and Guestrin shows the formula for the objective function

$$L^{(t)} = \sum_{i=1}^{n} (l (y_i, \hat{y}_i^{(t-1)} + f_t(x_i) + \Omega(f_t) \quad (7)$$

where n is the number of examples, m is the number of features, t is the number of leaves in each tree, ff relates to the independent tree structure and leaf, l is the differentiable convex loss function, $\hat{y}_i(t)$ be the prediction of the i-th instance at the t-th iteration, Ω is the penalized term (Chen & Guestrin, 2016). It is reasonable to not go too deep into the formula but only have a general idea of how XGBoost operates.

Some features of XGBoost are used, including (a) scalability– a highly scalable system due to several systems and algorithmic optimizations; (b) regularization–XGBoost includes different regularization penalties (e.g., L1 and L2) to prevent overfitting; (c) parallelization–that can be trained with multiple CPU cores; and (d) Non-linearity that the methodology can detect non-linear data patterns and learn from them.

XGBoost has been known for its high speed and high performance: XGBoost regression helps pick the most suitable machine learning model, which reduces time and enhances the model's accuracy. The methodology comes with an easy-to-read and interpret algorithm, and the algorithm's parameters are also easy to handle. Another highlight of XGBoost is that it handles missing values internally without a pre-treatment. The limitation of XGBoost is that it is very sensitive to outliers. Nevertheless, since all the parameters went through a cleaning process in the data cleaning phase, the outliers were either eliminated or replaced at this point of the project. On the other hand, XGBoost is recommended with the number of features that are less than 100 and less than the number of training samples, which is also satisfied in this section as we have 26 features (including the target variable), and the training set has 2409 rows. XGBoost also should not be used with image recognition and computer vision.

Other machine learning methodologies that are applicable to this project include Gradient Boosting, Random Forest, Time Series, etc.

Similar to XGBoost, Gradient Boosting is a technique that combines the prediction from a variety of decision trees and can be used in classification and regression tasks. Gradient Boosting also contains some parameters that XGBoost performs, such as ntrees, max_depth, learn_rate, col_sample_rate, etc. XGBoost is a more regularized version of Gradient Boosting: the Model Proposal section mentioned that one of XGBoost's purposes is to optimize the

regularization function, which lessens the overfitting situation. Furthermore, XGBoost is often faster than Gradient Boosting.

Another approach is using time series methodologies such as the autoregressive integrated moving average (ARIMA) that are useful in predicting the value of a continuous variable in the future. However, time-series predictive models usually require only one variable, and the prediction of the future values is based on the past values of that variable. For this project, as there are a variety of variables collected, it is more appropriate to choose a model that can utilize all the available variables. Furthermore, determining the most important features based on the machine learning model is also a part of the project: using time series forecasting requires picking only one feature at the beginning to process.

Another applicable model is Random Forest. Random Forest is defined as "a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest" (Breiman, 2001, p.5). Both Random Forest and XGBoost are ensemble learners, meaning a final model is developed on a collection of individual models. When compared with the Random Forest algorithm, XGBoost only needs very few hyperparameters. Random Forest will not overfit if the data is preprocessed and cleaned which our finalized dataset qualifies; this is a huge advantage for XGBoost if the data is not clean as XGBoost can automatically fill in the missing values. In general, Random Forest is a worth-to-try methodology for this project.

Random Forest Model.

As stated above, decision trees such as Random Forest have traits that would allow the algorithm to work with our current data. This also opens up the idea of potentially looking at other decision tree algorithms as well. For example, Nhu et al. (2020) discussed the comparison

of different decision tree algorithms such as “M5P, Random Forest, Random Tree, and Reduced Error Pruning Tree” in predicting lake water levels. (p. 1) They gathered a large assortment of data which contained a large number of variables that contributed to the rise and fall of the water level in the lake. As stated by Nhu et al. (2020) that one of the variables that was chosen is antecedent water level as one of the water sources for the lake is from springs in the area. (p. 3) The data was approached as a time series split by time category into different time lags to gain a better estimation of the impact of a random assortment of variables. From their research, they found that the M5P model had the highest coefficient of determination score and the lowest root mean squared error score followed closely by the Random Forest model. A limitation in the M5P model was that it was not as strong when it comes to datasets with smaller feature sets. It relied on having a large feature set to constantly decrease the overfitting. While the Random Forest model excelled in its versatility in dealing with all types of datasets, and does not purely rely on a large dataset to reduce overfitting. The approach to the prediction of reservoir storage was similar for both projects but the data collection between the research done by Nhu and their associated author's research article and the research for this project were different. Due to this difference, the choice to utilize the Random Forest model over the MP5 model came down to the versatility that Random Forest provided over MP5 in dealing with smaller datasets and features that did not have as high of importance to the dependent variable.

Another research article compared the Random Forest model and five other machine learning models against established reservoir water level prediction system models. The results of their research as discussed by Wang and Wang (2020) “the results indicate that ML models are superior in predictive accuracy compared to AHPS” (p. 1). In the results, they found that the Random Forest model scored higher than current prediction model systems and had a decently

accurate score. The AHPS model that they referenced was a “process-based advanced hydrologic prediction system” (Wang & Wang, 2020, p.1). The system takes in similar features such as meteorological data to predict water level. The approach and the data collected by Wang and Wang were similar to the approach and data utilized in this report. One of the small differences in their report is that they received most of their meteorological data in hourly increments instead of daily. Another small difference is that they were able to collect wind speeds, humidity, and longwave and shortwave radiation as a part of their meteorological data. The strengths of the prediction system as stated by Wang and Wang (2020) relied on “the models to represent the aquatic conditions” as well as “describing the variabilities in the observations” (p. 1) This would also be its limitations as it would rely on purely current meteorological data. The strength of the Random Forest model was that it utilized both current conditions and historical data to accurately predict the water levels. Another ML algorithm worthy to be tested to see if its effective in predicting reservoir levels is Gradient Boosting as multiple research papers tested comparisons between Gradient Boosting and Random Forest.

Gradient Boosting Model.

Due to the nature of the project limited the scope to using only one machine learning model per individual of the group. This removed the ability to compare the Gradient Boosting Model directly with the Random Forest model above with the data utilized in this project. Since no comparison can be made on the data, a comparison can still be made to determine which model is most compatible. This comparison was done utilizing the research papers found to compare this machine learning model.

Tian (2021) “compared random forests, gradient, boosting machine, extreme learning machine, M5-cubist, elastic net, as well as their multi-model ensemble using Bayesian

model averaging (BMA)" (p. 1) in their research to forecast reservoir inflow. Tian (2021) found that the Gradient Boosting Model for the Navajo reservoir "showed the best overall performance (KGE=0.76, R=0.83)" (p.12). The Navajo reservoir water supply chain is also very similar to Santa Clara County in that "the Navajo reservoir is located in the UCRB basin in the western US which is dominated by snow" (Tian, 2021). Santa Clara County's water supply chain is also dominated by snow via the Sierra snowpack. The Gradient Boosting Model was found to be best compared to all the other machine learning models when it had all variables as the input.

The local data used for this project is known to have data gaps or inconsistency. Caused by environmental factors such as loss of signal on the sensors or man-made factors like a government shutdown. This would require a greater importance to adaptability to such inconsistency or gaps. To resolve this issue, interpolation and extrapolation of the data will be needed. In Shuang's (2021) research on machine learning models predicting water demand, he found that the results "suggest that the gradient boosting decision tree (GBDT) model demonstrates the best prediction performance in the two scenarios" (p. 1) of interpolation and extrapolation.

Gradient Boosting has shown in both documentation and active research to be very tolerant in the data quality used for the model, especially volatile data. With Tian (2021) research parameters being very similar to the water supply chain found in Santa Clara County and Shuang's (2021) suggestion of the flexibility in Gradient Boosting, this model showed great promise in being able to work with the local parameters found in Santa Clara County at this point in time. Another model that also showed great promise is SVR as the testing utilizing time series data showed that SVR could have high accuracy.

SVR Model.

In this section, comparisons are made between the SVR model and one of the models Hipni et al. did for their research about dam water level predictions in 2013. Justification for the SVR model is also made in this section. In Hipni et al. research (2013), they compared the predicted results between the SVR model and the adaptive neuro fuzzy inference system (ANFIS) model. In the SVR model they did, they provided predictions using two different types of regression, SVR with loss function and Nu SVR, and there were three different scenarios considered to be the input data for each specific type of SVR. The inputs for their SVR model are (a) the daily rainfall for the past four days; (b) the dam water level for the past four days; and (c) the average of dam water level for the past seven days. The three scenarios are: (a) by itself only, (a) and (c), and (a) and (b), respectively. They concluded that the third scenario produced the best results. However, the comparisons made between this and Hipni et al.'s model does not use the third scenario, but the second one.

The second scenario using the first type of regression is compared to the second scenario in my model. In using Hipni et al. model, my model uses water storage level instead of dam water level and snow water equivalent data instead of rainfall. Majority of the water comes from the snow in the Sierra area, which corresponds to the rainfall in Hipni et al. model. My model uses aggregated data for the past seven and 30 days and Hipni et al.'s model uses aggregated data for the past four and seven days. They both have similar output which is the water level of the study area. Furthermore, both models employ the same SVM and share two similar metrics to evaluate the model. Both models employ SVR that incorporates a loss function and both did 10-fold cross validation for the model. Table 3 shows an overview of the similarities and differences between the models.

Table 3

Comparisons between This Model and Hipni et al. Model

		This Model	Hipni et al. Model
Inputs	Features Used	Water Storage Level	Dam Water Level
		Snow Water Equivalent	Rainfall
	Time Duration	Seven Days and 30 Days	Four Days and Seven Days
Outputs	Target Variable	Water Storage Level	Dam Water Level
Model	Algorithm	SVR	SVR
Evaluation	Metrics	R2, RMSE, and MAE	R, RMSE, MAPE, and MAE

Since both models use RMSE and MAE to evaluate the model, the results of these two metrics are retrieved and shown in Table 4. Though the RMSE and MAE values for my model are small, Hipni et al.'s RMSE and MAE values are even smaller, which could indicate that there are possible improvements that can be made with my model.

Table 4

Comparisons on Prediction Results between This Model and Hipni et al. Model

Model	This Model	Hipni et al. Model
Scenario	II	II
RMSE	0.00778	0.0000147
MAE	0.00591	0.000001

The type of the problem we are solving for this model is a regression problem. The type of the algorithm for this model is supervised machine learning and the characteristic of the algorithm is SVR. SVR is suitable for this targeted problem because the historical water storage level and the snow water equivalent amount do not appear to be linear and SVR transforms the nonlinear data to a hyperdimensional space so that a linear plane that separates all the observed points can be found within the space. Another advantage of SVR, mentioned by Khan and Coulibaly (2006), is that SVR computes efficiently. Therefore it saves memory and it does not occupy CPU largely. Khan and Coulibaly also mentions that SVR training ensures the search of the global minimum of the surface of the difference between the observed and the predicted values. However, Khan and Coulibaly state that SVR suffers from a longer training time for a dataset that is huge, but since we do not have a large dataset, the training time is effectively short for this model.

In reality, it is common that data is incomplete or insufficient to predict an accurate result. In this model, because there are only two main input features for the model, each of the input features is critically important for the model, especially the historical water storage level data.

Therefore, having a high complete and sufficient ratio of the historical water storage level data is necessary which also implies the incompleteness and insufficiencies of the data would jeopardize the outcome of the model. On the other hand, because how input data of this model is aggregated to the most recent time period, the model gets to learn the data and produce predictions in a very efficient way. The model can even produce relatively accurate results picking up sharp increases and steep drops in the trend with very limited years of data, which is further discussed in detail in the Model Validation and Evaluation section. However, the predicting period of this model is only one month ahead which means if the water management team is seeking a prediction for a further future, this model would not be suitable for them. The model is also not automated meaning upcoming new data needs to be added to the model for training manually.

4.4 Model Evaluation Methods

In this section, model evaluation methods are described. Coefficient of determination score (R^2), root mean square error ($RMSE$) are used for all models. Additional to the two, mean square error (MSE) is used for XGBoosting, Gradient Boosting, and Random Forest models and , mean absolute error (MAE), and the mean error (ME) in percentage are used for the SVR model. 10-fold cross validation is performed for all models too.

First, the model is evaluated by plotting out the predicted points and the observed points. If the predicted points are too far from the observed points, that means the predictions are very inaccurate and we would need to go back to fix the model. If the predicted points are very close to the observed points, four metrics are calculated to determine how good the model performs, including coefficient of determination score (R^2), mean square error (MSE), root mean square error ($RMSE$), mean absolute error (MAE), and the mean error (ME) in percentage.

R^2 is calculated as

$$R^2 = 1 - \frac{RSS}{TSS} \tag{8}$$

where RSS represents the residual sum of squares and TSS represents the total sum of squares. MSE is calculated as

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{N} \tag{9}$$

where y_i represents the actual target value, $\hat{y}_i$ represents the predicted value, and N is the total number of predictions made.

$RMSE$ is calculated as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{N}} \tag{10}$$

where y_i represents the actual target value, $\hat{y}_i$ represents the predicted value, and N is the total number of predictions made.

MAE is calculated as

$$MAE = \frac{1}{N}\sum_{i=1}^{n} \left| y_i - \hat{y}_i \right| \tag{11}$$

where y_i represents the actual target value and $\hat{y}_i$ represents the predicted value, and N is the total number of predictions made. ME in percentage is calculated as

$$ME = \frac{1}{N}\sum_{i=1}^{n} (y_i - \hat{y}_i) \tag{12}$$

where y_i represents the actual target value, $\hat{y}_i$ represents the predicted value, and N represents the total number of predictions made. R^2 is usually between zero to one. We would like R^2 to be as close as possible to one and $RMSE$, MAE, and ME value to be as close as possible to zero

For XGBoosting, repeated k-fold cross validation is also performed. Repeated k-fold cross validation means running k-fold cross validation several times to possibly obtain a more reliable performance. Just like k-fold cross validation, one disadvantage of Repeated K-Fold Cross-Validation is that it can overlap the data between each round. The cross-validation function is available in both XGBoost and Sklearn packages.

Graphs with the actual and predicted values for each scenario are also presented to visualize how well the corresponding model performs as well as k-fold cross validation to make sure all possible input data can be predicted correctly. The data is also split by different periods of time to see how the prediction reacts to different sizes of training and testing dataset.

4.5 Model Validation and Evaluation

For XGBoosting, Gradient Boosting, and Random Forest Model.

XGBoosting Model.

After splitting, the training dataset of the independent variables, X_train, has 2409 rows and 25 columns, the testing dataset of the independent variables, X_test, has 603 rows and 25 columns. The training score (R^2) is 0.68, which indicates that the model is nearly identical with the training datasets.

The average K-fold Cross Validation (n_split = 10) score gives a lower score of 48.73% and average K-fold Cross Validation (n_split = 10) repeating (three times) score is 49.49%.

The next step is to use the model and the testing dataset to predict the water storage value. After obtaining the prediction values, find the testing score (R^2) between those values and

the actual values in the testing dataset. The testing score is around 0.53 in Figure 8. Notice that although the training score is higher than the testing score, the difference between them is not tremendous and in an acceptable range.

On the other hand, the MSE and RMSE between the prediction value and the actual value in the testing dataset are shown in the Figure 52 below. The MSE turns out to be small, about 0.04, and the RMSE is a little higher, 0.19.

Figure 52

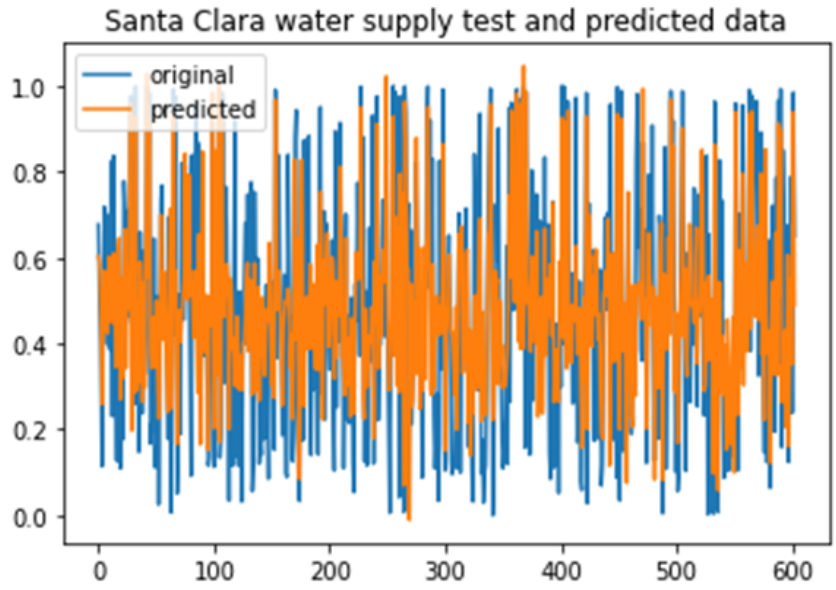
MSE and RMSE of The Prediction Value and Actual Value in The Testing Dataset.

```
MSE: 0.04
RMSE: 0.190324
```

The chart below illustrates the line chart of original water supply versus the predicted water supply in Stanislaus County. The x axis presents the number of records (603) which are shuffled in the training and testing splitting process. Figure 53 below shows similar trends in original and predicted lines. Observe that the prediction does not fluctuate as dramatically as the real data.

Figure 53

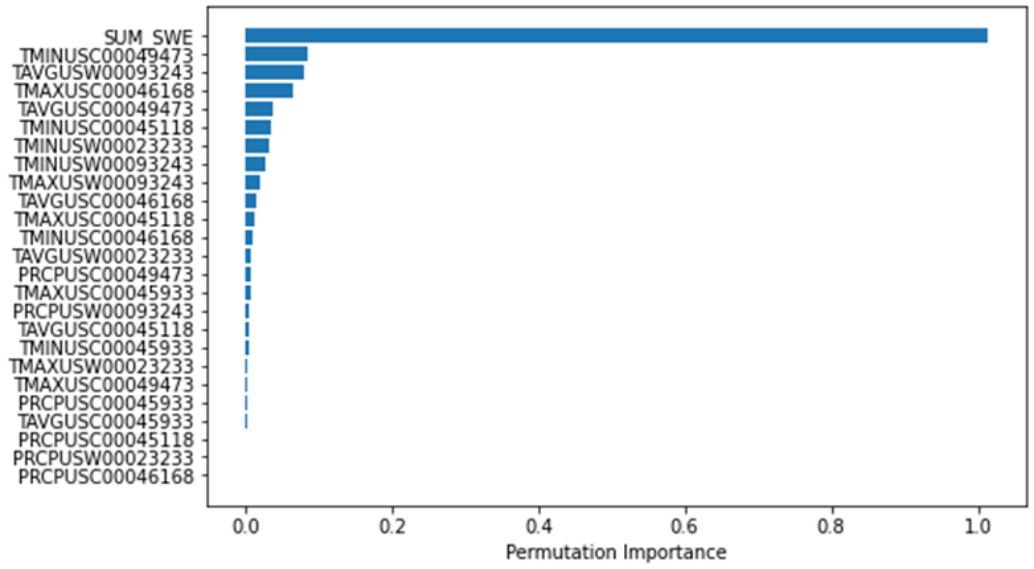
Testing and Predicted Data Comparison



The next data visualization (Figure 54) shows the feature importance when fitting the prediction model into the testing datasets by function perm_importance(). Observe that the daily amount of snow water in California is the most important factor of the XGBoost model that has been developed. Additionally, the daily minimum temperature of station ID USC00049473 and the average temperature in station ID USW00093243 are the second and third important features. Station with the ID USC00049473 is Watsonville Waterworks, CA US, and station with the ID USW00093243 is Merced 23 WSW, CA US.

Figure 54

XGBoost Water Supply Prediction in Santa Clara County Important Features



Usually, an R^2 score of 0.7 or more is considered good, 0.4 to 0.7 is considered average, and below 0.4 is considered poor. It is difficult to conclude if XGBoost Regression is a good fit model for this project. Based on the Coefficient of Determination alone, the model does not appear to perform well as the R^2 score falls into the average range. However, with low MSE and RMSE scores, XGBoost Regression is still a potential methodology.

Gradient Boosting.

As stated in the Model Support section, our model validation will be done through K-Fold cross validation on the training set. To achieve proper validation and evaluation, the Test data has to be independent of any form of training or manipulation on it. This is achieved by splitting the Labeled data into Train and Test with Test size being 20%. The Train data, which is 80% of the total data, gets validated through K-Fold cross validation on a 10-fold setting. The Train data subset 10 times, 10-fold, Figure 55 provides a visual summary on how the data is broken up. Each subset contains the same data but the validation set uses changes in each

iteration. Sklearn.model_selection.Kfold (2022) document states that k-1 of the remaining fold is used as the training set while the remainder is used as the validation. Figure 56 provides a visualization on how K-Fold works in this model.

Sanjay (2022) explains that “K-Folds technique is popular and easy to understand, it generally results in a less biased model compared to other methods”. One of the greatest strengths with K-Folds is in its ability to ensure all observation of the original dataset appears in both the training and test/validation set. Through my understanding of how K-Fold works we reasoned this was the best cross validation method for my model because my model has a limited amount of input data and the evaluation would benefit greatly as it is assumed that all parameters are important for regression.

Figure 55

Break-up of dataset for machine learning Model and Validation

<table>
<tr><td colspan="3">Labeled Data-100%</td></tr>
<tr><td colspan="2">Train-80%</td><td>Test-20%</td></tr>
<tr><td>Train – 90%</td><td>Validation – 10%</td><td>Test – 20%</td></tr>
</table>

Figure 56

Example visualization on K-Fold with 3 folds. Repeats the dataset 3 times but each repetition has a different validation selection and train selection (k-1)

Train	Validate	Train
Train	Train	Validate
Validate	Train	Train

Scikit-learn has two tools to help in validation under their model_selection module, cross_val_score and KFold. KFold is a quick and efficient way to set-up the parameter needed for KFold and run the code. Rather than writing a bunch of code to subset out the data and selecting one data to be the validation, KFold from scikit-learn does this for us in a simple format of KFold (n_splits=,shuffle=). Where n_splits represent the amount folds we want in my validation process. KFold from scikit-learn only sets up the validation system and doesn't actually run the process to produce a score. This is where cross_val_score comes in. Cross_val_score from scikit-learn is a tool to run a cross validation framework but it requires the user to set-up the framework beforehand. We did this earlier by utilizing KFold from scikit-learn. Figure 57 shows how this is set-up in Python.

Figure 57

Python script to create cross validation model KFold and implementing it to obtain cross validation score

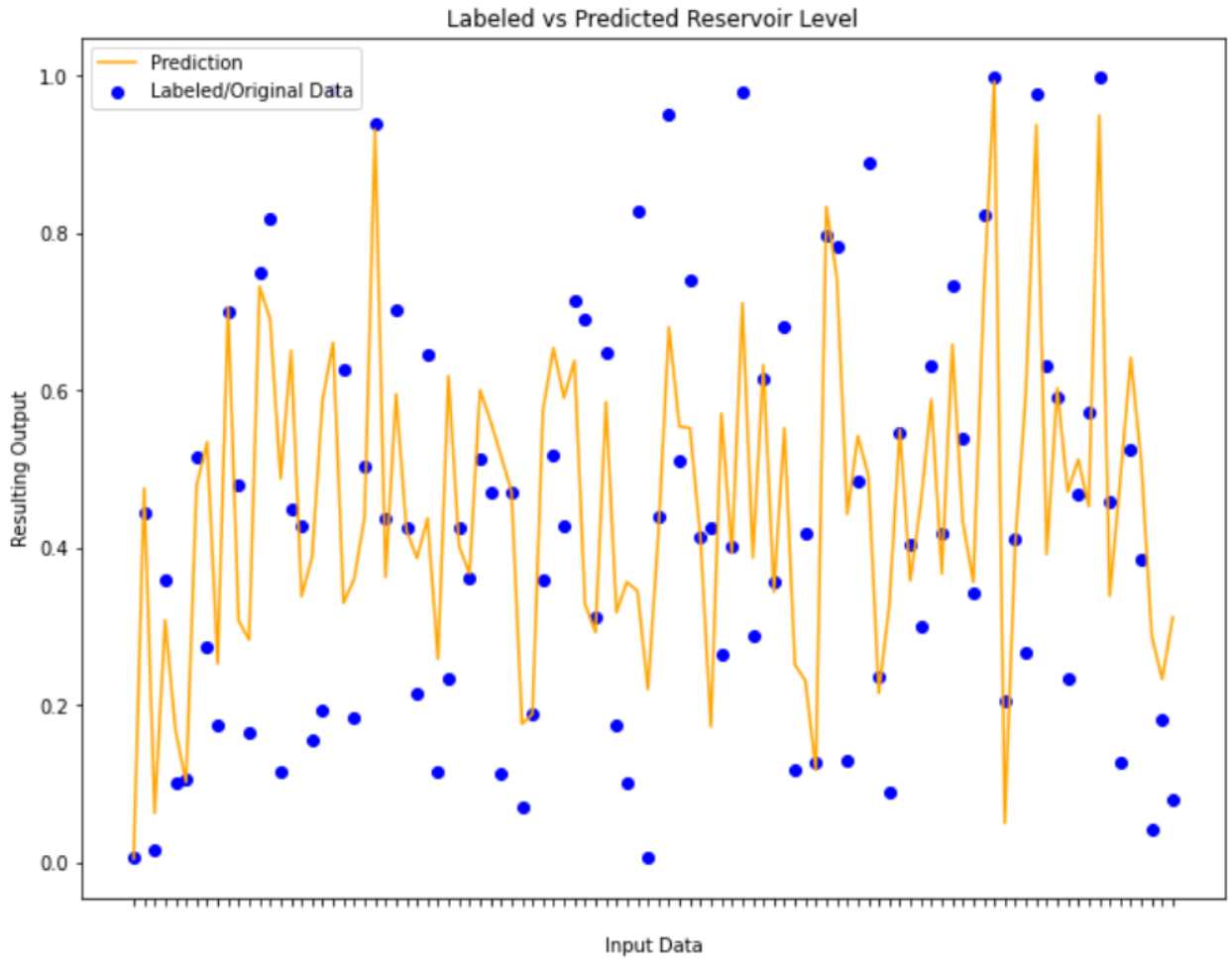
```
#validation K-fold
kfold=KFold(n_splits=10, shuffle=True)
kf_cv_score=cross_val_score(reg, X_train, y_train,cv=kfold)
kf_cv_score
```

Shuffle was also turned on in KFold to guarantee that the temporal aspect of the original dataset was removed and similar to the context of how the Test data would be used. Each iteration of K-Fold (total dataset split by 10) produced a score and to obtain something beneficial we found the average on all of the cross-validation score. The average of all the cross-validation score gavea score of 0.53. This lines up with my R^2 score on the Test dataset in relation to the predicted values.

To better illustrate these results we created a graph to plot the results. The labeled data we have is set as a scattered plot while the predicted values are set to be a line graph as shown in Figure 58.

Figure 58

Graph Showing How Often Predicted Values Intersect with Labeled Data



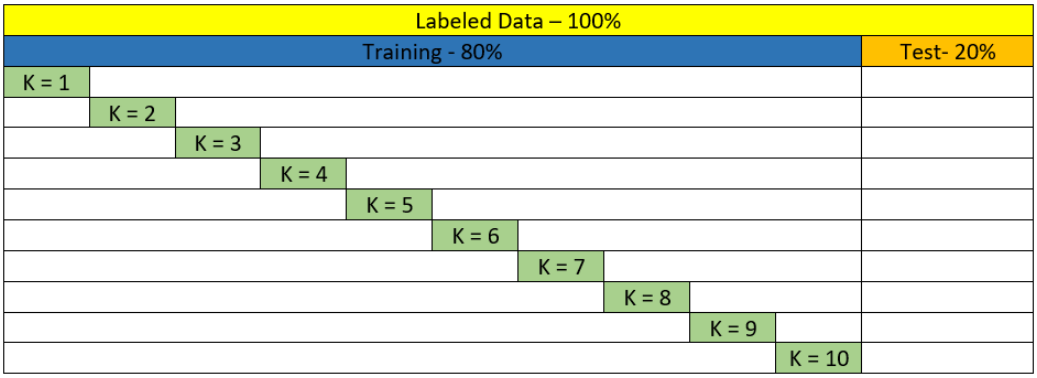
This is not a time series graph and a line graph for prediction was used to make it easier to relay when each point in the prediction had the same exact input as the blue point, Labeled data. Each tick on the x-axis represents a random input on a set of input parameters into the machine learning model. The y-axis represents the result of the machine model running. Because it is a regression, we are looking at how close the line is to the points. In my observation we found that the predicted value is close or exactly like the labeled data about half the time. This is in line with the evaluation and validation of my machine learning model.

Random Forest

The Random Forest model provided a viable machine learning prediction of a local reservoir. Figure 59 shows a breakdown of how the cross-validation worked on the training dataset. As well as the division of data for the training and testing data subsets.

Figure 59

The division of data for cross validation



The model scored a 0.6 coefficient of determination on the testing data which was not very high. The closer a value was to one the better the model was at predicting an approximate value close to the actual values. The training coefficient of determination was around 0.94 which tells us that the model excelled in the training dataset. The cross-validation score was around 0.56 which tells us that the data was overfitting the training data. This helped to explain the fairly low accuracy rate of the coefficient of determination score on the test data set. The mean absolute error score was around 0.13 which told us that the margin of error for the predicted values was on average 0.13 from the regression line. The mean square error and the root mean square error were 0.03 and 0.17 respectively. The mean square error was used to assist in finding the root mean square error. What the root mean square error represented was that the model

predicted within 0.17 of the actual normalized values. A results dataframe can be seen in Figure 60 which consists of 603 records.

Figure 60

The results dataframe

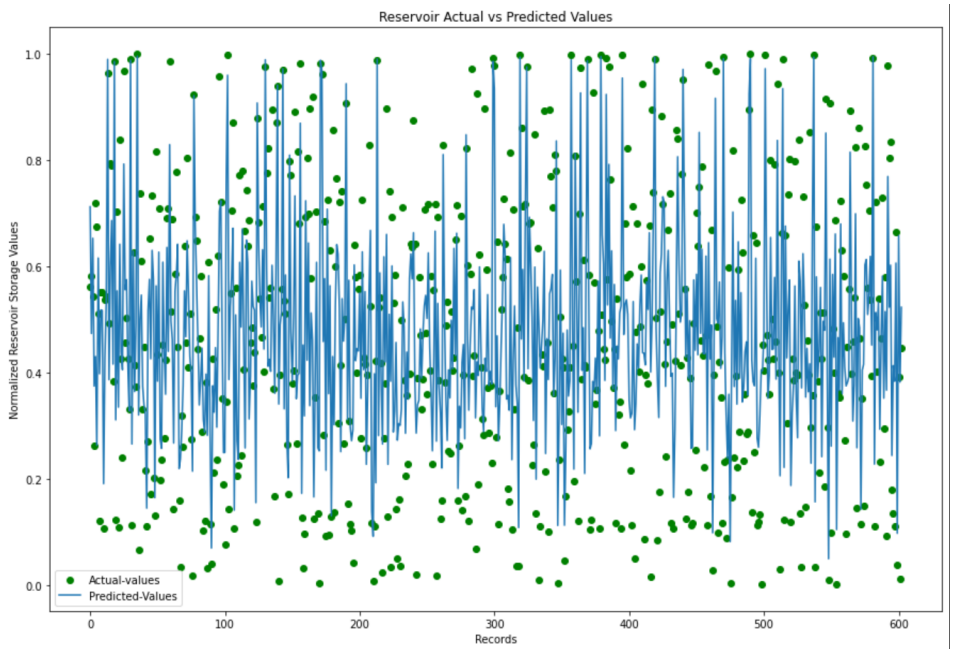
	Actual-Values	Predicted-Values
0	0.561499	0.712014
1	0.581165	0.473556
2	0.544229	0.652873
3	0.262367	0.374606
4	0.718535	0.430288
...	...	...
598	0.664324	0.606139
599	0.038343	0.097644
600	0.391793	0.657869
601	0.011985	0.382019
602	0.447259	0.522917

603 rows × 2 columns

The visualization of the predicted and actual values are shown in Figure 61. This visualization showed that the predicted values were not quite predicting values around the actual values. As we can see in several instances, the model predicted a value way smaller or way bigger than the actual values. This made up the bulk of the 40% of the values that were predicted incorrectly by the model. We can also see a large portion of the data points were along the predicted values line and were predicted fairly well.

Figure 61

Predicted vs Actual Values for Random Forest Model



As discussed in the previous section, the differences between this model and similar models done by researchers consisted of several key parameters. The first difference was in how the research papers collected their data and the features they were able to take out of the data. The ability to collect vast amounts of data from the raw data sources for river flow and meteorological data allowed them to have more data features to work with. This would give them the ability to identify more data features with higher feature importance to their target data. The second main difference was the feature importance of the data used for this project varied from extremely important to not important at all which can be seen in Figure sixty two. The lack of significantly important data features did not contribute much to supporting the model.

Figure 62

Feature Importance Graph

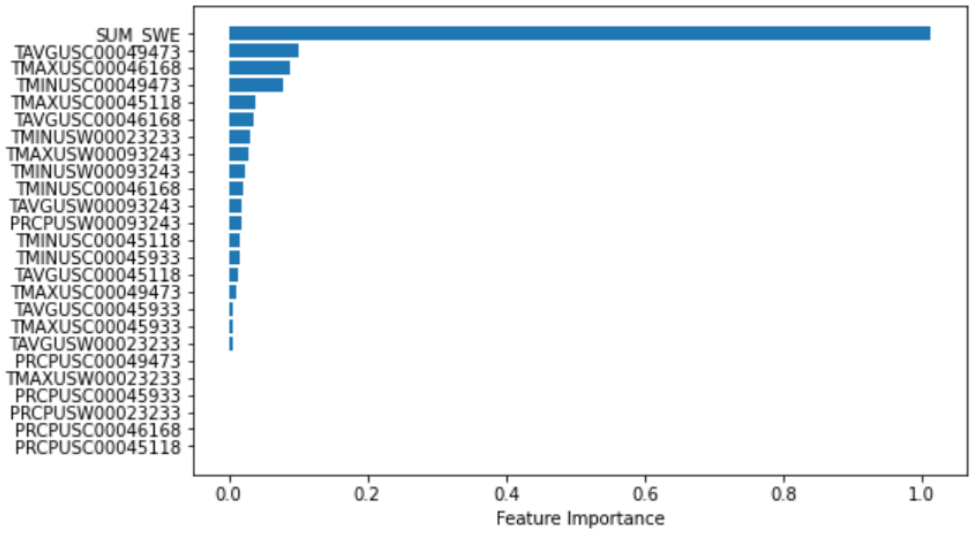


Table 10 shows a summary of the evaluation score we receive on the machine learning models that did not use MAE evaluation. With the same input of data, Extreme Gradient Boosting (XGBoost) model appears to have the highest R^2 score among the three models and Random Forest model has the smallest RMSE and MSE values.

Table 10

Evaluation Result for XGBoosting, Gradient Boosting, and Random Forest Model

Model	XGBoosting	Gradient Boosting	Random Forest
R^2	**0.68**	0.50449	0.60132
RMSE	0.19032	0.30083	**0.17393**
MSE	0.04	0.03154	**0.03025**

For the SVR Model.

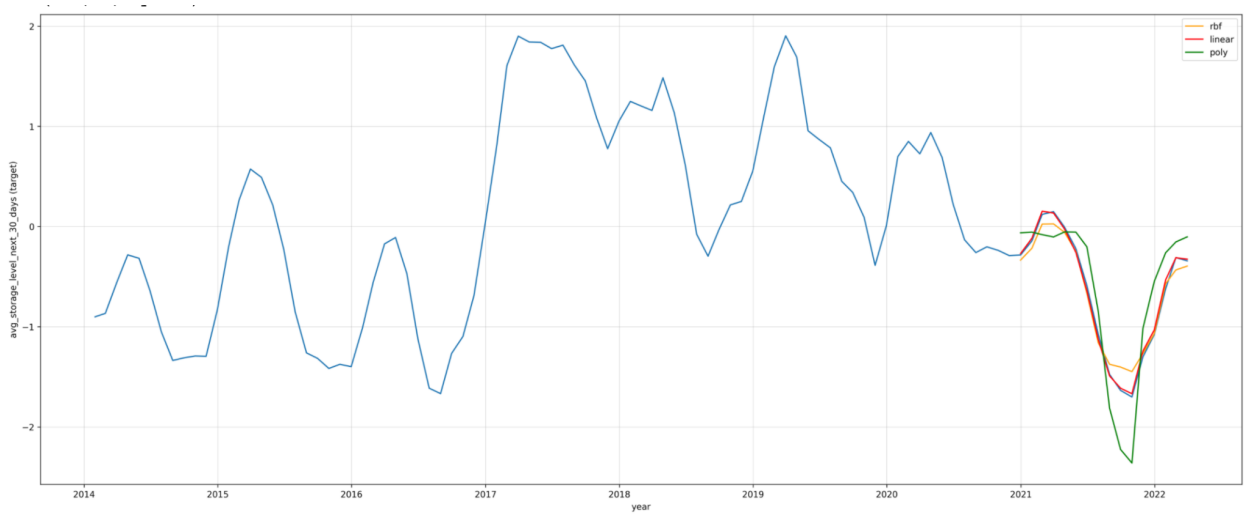
To visually look at the accuracy of the model for both scenarios, they appear to have very similar accuracy. Figure 63 shows the prediction results using the three kernel functions for scenario one and Figure 64 shows the prediction results using the three kernel functions for scenario two. The prediction accuracy looks very similar for both scenarios that we visually cannot tell which one is more accurate. To compare the kernel functions, linear kernel function appears to be the one that provides the most accurate predictions.

Figure 63

Prediction for Scenario we Presented in Line Chart

Figure 64

Prediction for Scenario II Presented f in Line Chart



For linear kernel function only, grid searches are used to tune its hyperparameters, and to make sure the model with the tuned hyperparameters works with all data we have, 10-fold cross validation is performed. Table 11 shows the best hyperparameter for both scenarios after 10-fold cross validation. For the second scenario, it seems the higher the C level is, the higher the R^2 it is, but to avoid overfitting and save computing time, C is set to 100 only.

Table 11

Tuned Hyperparameter Results for Both Scenarios

Scenario	I	II
Kernel Function	linear	linear
Regularization parameter, C	10	100
Insensitive loss parameter, ε	0.001	0.0001

Though Figure 63 and 64 show the linear kernel function predicts the results better, we cannot tell which scenario gives us a better result from the figures so we need the metrics mentioned in the Model Evaluation Methods to determine that, which are coefficient of determination score, root mean square error, mean absolute error, and the error percentage between the actual and the predicted values. Table 12 shows the evaluating metrics score for both of the scenarios with the tuned hyperparameters.The second scenario, which is the scenario with snow water equivalent data, appears to perform better than the first scenario.

Table 12

Evaluation Results of Both Scenarios

Scenario	I	II
R^2	0.99515	**0.99623**
RMSE	**0.04254**	0.30083
MAE	0.03551	**0.03154**
ME %	-2.79%	**-1.24%**

To validate what the best training and testing data size produces the best predictions, data is split by 16 times. The first splitting starts on January 1st 2021 and each time moves six months before the previous date. Each splitting goes through the modeling phase shown in Figure 50 to produce the best prediction. After 16 times splitting and training is done, a graph is presented. Figure 65 shows the 16 predictions for the second scenario with different sizes of training and testing data, where the orange line is the prediction and the blue line is the observed line. From left to right and from top to bottom, the training size decreases by six months and the testing size

increases by six months. The prediction performance overall drops as training data size decreases. However, that is not always true that the more training data does not always produce a better prediction. Figure 66 shows that seven years of training data give us the highest R^2 score and the lowest RMSE score which indicates the seven years of training data gives us the best prediction among all different training and testing sizes, and R^2 and RMSE score start to drop a little when training size goes more than seven years.

Figure 65

Prediction Comparisons on Different Training and Testing Sizes for The Second Scenario

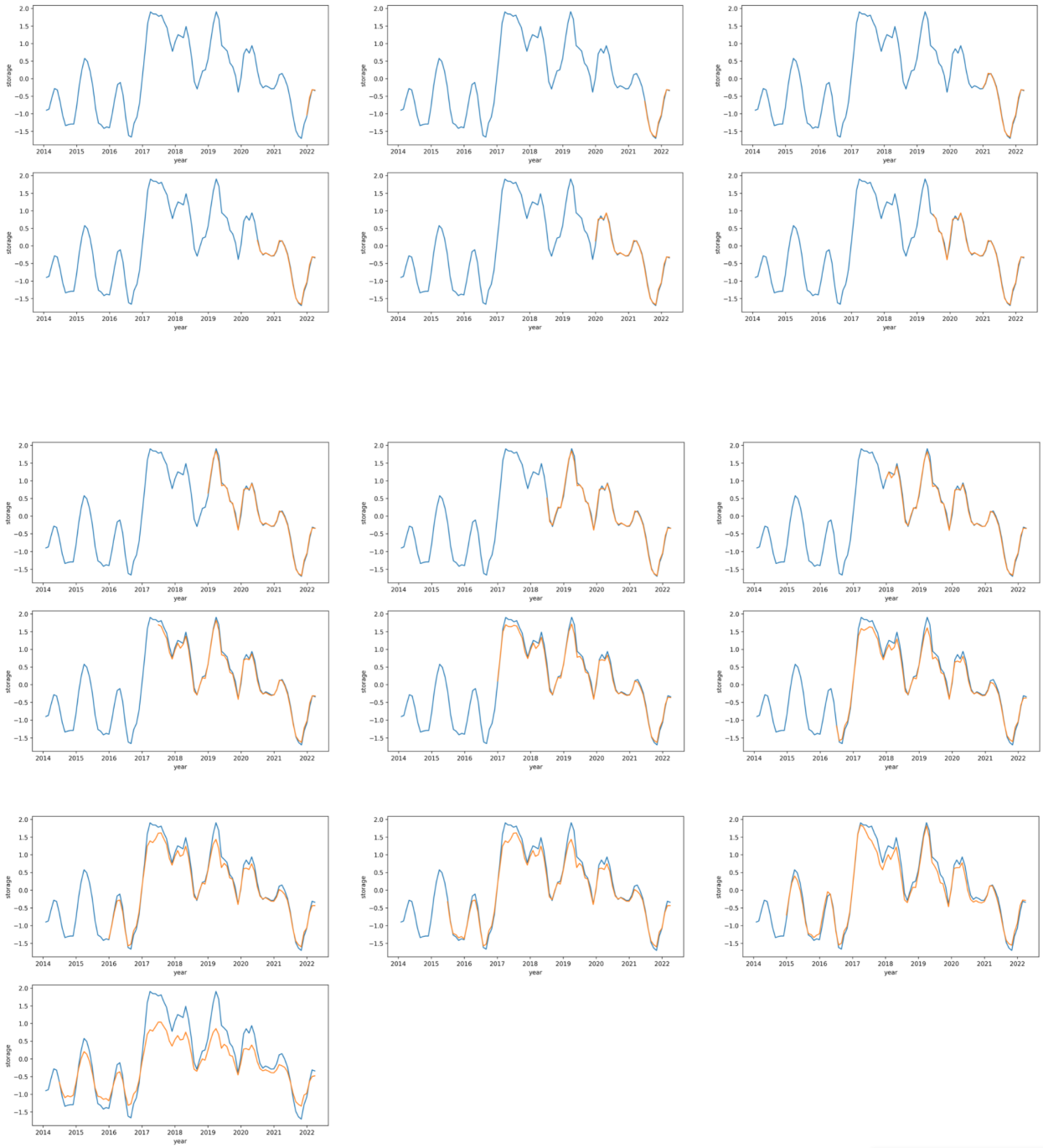
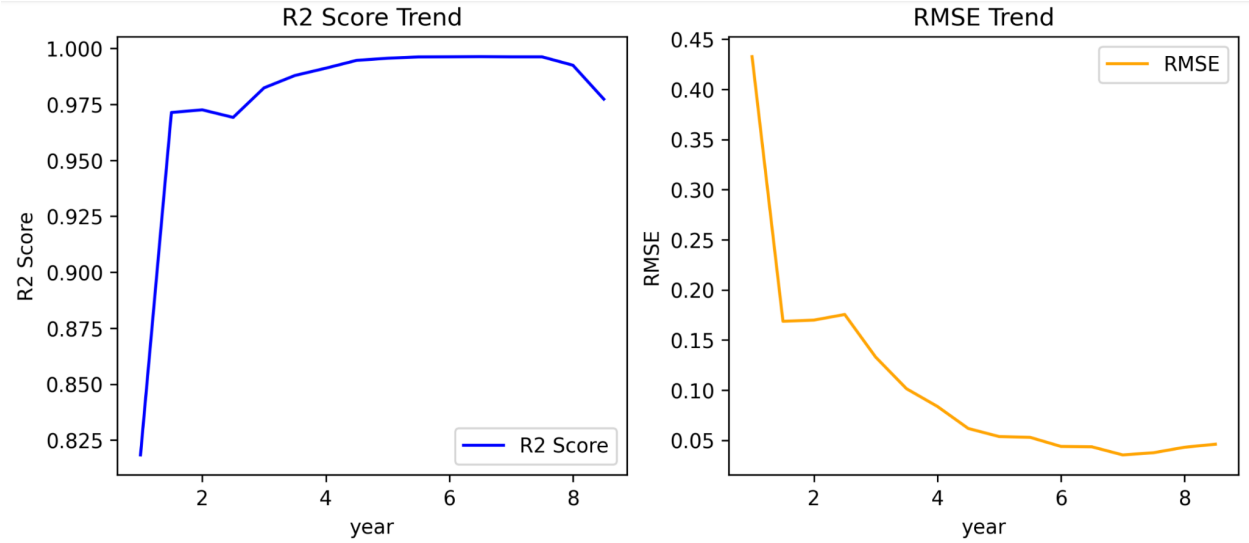


Figure 66

R^2 and RMSE Score Trend on Different Training and Testing Sizes for The Second Scenario



References

Ambrosio, J. K., Brentan, B. M., Herrera, M., Luvizotto, E., Ribeiro, L., & Izquierdo, J. (2019). Committee Machines for Hourly Water Demand Forecasting in Water Supply Systems. Mathematical Problems in Engineering, 2019, 1–11. https://doi.org/10.1155/2019/9765468

Breiman, L. (2001). Random Forests. Machine Learning 45, 5–32 https://doi.org/10.1023/A:1010933404324

California Department of Water Resources. (2022, April 5). *Daily SWP reservoir elevation and Storage Data*. California Natural Resources Agency Open Data. Retrieved April 22, 2022, from https://data.cnra.ca.gov/dataset/state-water-project-monthly-report-of-operations

California Water Boards. (2020, March). *Reservoir Storage Measurement & Recordkeeping Guide*. https://www.waterboards.ca.gov/waterrights/water_issues/programs/diversion_use/docs/res_measure6.pdf

Carbonneau, R., Laframboise, K., & Vahidov, R. (2008, February). Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, Volume 183, Issue 3, Pages 1140-1154. https://doi.org/10.1016/j.ejor.2006.12.004

Carvalho, T.M.N., de Assis de Souza Filho, F. (2021). Variational Mode Decomposition Hybridized With Gradient Boost Regression for Seasonal Forecast of Residential Water

Demand. *Water Resour Manage* 35, 3431–3445. https://doi.org/10.1007/s11269-021-02902-7

Castillo-Botón, C., Casillas-Pérez, D., Casanova-Mateo, C., Moreno-Saavedra, L. M., Morales-Díaz, B., Sanz-Justo, J., Gutiérrez, P. A., & Salcedo-Sanz, S. (2020). Analysis and prediction of dammed water level in a hydropower reservoir using machine learning and persistence-based techniques. *Water*, *12*(6), 1528. https://doi.org/10.3390/w12061528

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.2939785

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297. https://doi.org/10.1007/bf00994018

Delpla, I., Jung, A. V., Baures, E., Clement, M., & Thomas, O. (2009). Impacts of climate change on surface water quality in relation to drinking water production. *Environment International*, *35*(8), 1225–1233. https://doi.org/10.1016/j.envint.2009.07.001

Dodge, Yadolah. (2008) Coefficient of determination. (n.d.). *The Concise Encyclopedia of Statistics*, 88–91. https://doi.org/10.1007/978-0-387-32833-1_62

Elavarasan, D., Vincent, D. R., Sharma, V., Zomaya, A. Y., & Srinivasan, K. (2018). Forecasting yield by integrating agrarian factors and machine learning models: A survey. Computers and Electronics in Agriculture, 155, 257–282. https://doi.org/10.1016/j.compag.2018.10.024

Hipni, A., El-shafie, A., Najah, A., Karim, O. A., Hussain, A., & Mukhlisin, M. (2013). Daily Forecasting of Dam Water Levels: Comparing a Support Vector Machine (SVM) Model

With Adaptive Neuro Fuzzy Inference System (ANFIS). *Water Resources Management*, 27(10), 3803–3823. https://doi.org/10.1007/s11269-013-0382-4

Hussein, E. A., Thron, C., Ghaziasgar, M., Bagula, A., & Vaccari, M. (2020). Groundwater prediction using machine-learning tools. Algorithms, 13(11), 300. https://doi.org/10.3390/a13110300

Ibrahem Ahmed Osman, A., Najah Ahmed, A., Chow, M. F., Feng Huang, Y., & El-Shafie, A. (2021). Extreme gradient boosting (xgboost) model to predict the groundwater levels in Selangor Malaysia. *Ain Shams Engineering Journal*, *12*(2), 1545–1556. https://doi.org/10.1016/j.asej.2020.11.011

Imported Water: Vital to Santa Clara County. *Santa Clara Valley Water*, https://www.valleywater.org/your-water/where-your-water-comes/imported-water.

Jiang, M., Jiang, S., Zhu, L., Wang, Y., Huang, W., & Zhang, H. (2013). Study on Parameter Optimization for Support Vector Regression in Solving the Inverse ECG Problem. *Computational and Mathematical Methods in Medicine*, *2013*, 1–9. https://doi.org/10.1155/2013/158056

Khan, M. S., & Coulibaly, P. (2006a). Application of Support Vector Machine in Lake Water Level Prediction. *Journal of Hydrologic Engineering*, *11*(3), 199–205. https://doi.org/10.1061/(asce)1084-0699(2006)11:3(199)

Kilimci, Z., Akyuz, A., Uysal, M., Akyokus, S., Uysal, M., Bulbul, B., & Ekmis, M. (2019). An Improved Demand Forecasting Model Using Deep Learning Approach and Proposed Decision Integration Strategy for Supply Chain. Complexity. 9067367(2019). 1-15. https://doi.org/10.1155/2019/9067367

Kumari, A. (2022, April 11). *Mean squared error or r-squared - which one to use?* Data Analytics. Retrieved May 12, 2022, from https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/

Miro, M. E., Groves, D., Tincher, B., Syme, J., Tanverakul, S., & Catt, D. (2021). Adaptive water management in the face of uncertainty: Integrating machine learning, groundwater modeling and robust decision making. *Climate Risk Management*, *34*, 100383. https://doi.org/10.1016/j.crm.2021.100383

National Centers for Environmental Information (NCEI). (2022, April 12). *About*. Retrieved April 12, 2022, from https://www.ncei.noaa.gov/about

National Centers for Environmental Information (NCEI). (n.d.). *Global Historical Climatology Network - daily (GHCN-daily), version 3*. Dataset Overview | National Centers for Environmental Information (NCEI). Retrieved April 22, 2022, from http://doi.org/10.7289/V5D21VHZ

Nhu, V.-H., Shahabi, H., Nohani, E., Shirzadi, A., Al-Ansari, N., Bahrami, S., Miraki, S., Geertsema, M., & Nguyen, H. (2020). Daily water level prediction of Zrebar Lake (Iran): A comparison between m5p, random forest, random tree and reduced error pruning trees algorithms. *ISPRS International Journal of Geo-Information*, *9*(8), 479. https://doi.org/10.3390/ijgi9080479

Pathak, T., Maskey, M., Dahlberg, J., Kearns, F., Bali, K., & Zaccaria, D. (2018). Climate Change Trends and Impacts on California Agriculture: A Detailed Review. *Agronomy*, *8*(3), 25. https://doi.org/10.3390/agronomy8030025

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel , O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas , J., Passos, A., Cournapeau , D.,

Brucher, M., Perrot, M., & Duchesnay, E., (2011), Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

Ponraj, A. S., & Vigneswaran, T. (2019). Daily evapotranspiration prediction using gradient boost regression model for irrigation planning. *The Journal of Supercomputing*, *76* (8), 5732–5744. https://doi.org/10.1007/s11227-019-02965-9

Refaeilzadeh P., Tang L., Liu H. (2009) Cross-Validation. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_565

Sahoo, S., Russo, T. A., Elliott, J., & Foster, I. (2017). Machine learning algorithms for modeling groundwater level changes in agricultural regions of the U.S. *Water Resources Research*, *53*(5), 3878–3895. https://doi.org/10.1002/2016wr019933

Sanjay, M. (2020, August 19). *Why and how to cross validate a Model?* Medium. Retrieved May 13, 2022, from https://towardsdatascience.com/why-and-how-to-cross-validate-a-model-d6424b45261f

Sapitang, M., M. Ridwan, W., Faizal Kushiar, K., Najah Ahmed, A., & El-Shafie, A. (2020). Machine learning application in reservoir water level forecasting for Sustainable Hydropower Generation Strategy. *Sustainability*, *12*(15), 6121. https://doi.org/10.3390/su12156121

Seyedan, M., & Mafakheri, F. (2020). Predictive big data analytics for supply chain demand forecasting: methods, applications, and research opportunities. *Journal of Big Data*. *7*(53). 1-22. https://doi.org/10.1186/s40537-020-00329-2

Shen, Z., & Yong, B. (2021). Downscaling the GPM-based satellite precipitation retrievals using gradient boosting decision tree approach over Mainland China. *Journal of Hydrology*, *602*, 126803. https://doi.org/10.1016/j.jhydrol.2021.126803

Shuang, Q., & Zhao, R. T. (2021). Water Demand Prediction Using Machine Learning Methods: A Case Study of the Beijing–Tianjin–Hebei Region in China. *Water*, *13*(3), 310. https://doi.org/10.3390/w13030310

Singh, H. (2018, November 4). *Understanding gradient boosting machines*. Medium. Retrieved May 13, 2022, from https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab

Sklearn.ensemble.GradientBoostingRegressor (n.d.). Retrieved May 12, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

Sklearn.ensemble.randomforestregressor. scikit. (2022). Retrieved May 14, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Sklearn.metrics.explained_variance_score. scikit. (n.d.). Retrieved May 12, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.explained_variance_score.html

Sklearn.metrics.r2_score. scikit. (n.d.). Retrieved May 12, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

Sklearn.model_selection.Kfold. scikit. (n.d.). Retrieved May 13, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

Sklearn.model_selection.train_test_split. scikit. (2022). Retrieved May 13, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, *14*(3), 199–222. https://doi.org/10.1023/b:stco.0000035301.49549.88

South Bay Water Officials Draw on Distant Reserves, Weigh Options as Drought Deepens. *CBS News*, CBS Interactive, 30 Apr. 2021, https://www.cbsnews.com/sanfrancisco/news/california-drought-south-bay-water-officials-distant-reserves-increased-storage-water-recycling/.

Tian, D., He, X., Srivastava, P., & Kalin, L. (2021). A hybrid framework for forecasting monthly reservoir inflow based on machine learning techniques with dynamic climate forecasts, satellite-based data, and climate phenomenon information. *Stochastic Environmental Research and Risk Assessment*. https://doi.org/10.1007/s00477-021-02023-y

United States Department of Agriculture. (n.d.). *Natural Resources Conservation Service*. Automated Snow Monitoring. Retrieved April 13, 2022, from https://www.nrcs.usda.gov/wps/portal/wcc/home/aboutUs/monitoringPrograms/automatedSnowMonitoring/

Wang, Q., & Wang, S. (2020). Machine learning-based water level prediction in Lake Erie. *Water*, *12*(10), 2654. https://doi.org/10.3390/w12102654

Zhang, F., & O'Donnell, L. J. (2020). Support vector regression. *Machine Learning*, 123–140. https://doi.org/10.1016/b978-0-12-815739-8.00007-9

Appendix

Source Code

Github link: https://github.com/hoangkhanhnghi01/Data270Project