# STAT 450 Individual Report

The Effects of Climate Variables on Average Stream Flow for Canadian Watersheds

By Anjali Chauhan (57181489)

Feb 20, 2022

## 1. Summary

In this analysis, we sought to develop a model that predicts the annual average streamflow for Canadian watersheds by studying the effects of climate variables on the annual average streamflow to provide crucial information for efficient water resource management. This helps in reducing the economic and financial loss due to floods, droughts and dam mismanagement. By training an ensemble of prediction models like Radial SVM, Lasso and Ridge Regression, Random Forest, GBM, etc, with important variables selected through several variables importance techniques like RFE, Exhaustive Search, etc and predicting the annual average streamflow, we achieved a high prediction performance with the lowest RMSE of 0.179. These techniques with further analysis can be broadly applied to predict average annual stream flow across different watersheds throughout the planet without the spatial and temporal input.

## 2. Introduction

A watershed refers to an area of land where rainfall and snowmelt collect and stream into a common outlet like rivers, lakes and other bodies of water. Understanding watershed stream flow is important for water resource management e.g. irrigation, hydroelectric power and flood control. The study aims to investigate and understand the effect of climate variables on the watershed's streamflow.

The analysis aims to address the ~~client's research~~ questions:

- Can the data from one catchment be used to extrapolate stream flow in another catchment given the climate variables?
- Can we detect the unusual streamflow activity accurately that can lead to severe adverse effects for the nearby ecosystem and populated areas?

More specifically, the analysis has the following objectives to answer the above questions:

- To build an anomaly detection model to predict the unusual streamflow activity with low false positives and negatives.
- To translate the relationship between the response and multiple predictors, measured over many years, into insightful visualizations
- To ~~obtain the most~~ important climate variable(s) and determine the effect of said variable(s) on the streamflow
- To model and predict the average stream flow values beyond 2018

This report summarizes all of the primary statistical modeling and analysis results associated with the ~~client's~~ study. The remainder of this report is organized as follows: Section 3 describes the data collection, provides measurement of the variables and summarizes the data. Section 4 presents the statistical modelling techniques used to answer the client's research questions. Section 5 summarizes and interprets the results of the statistical analysis conducted. Appendices are provided for further exploratory data analysis along with the code used for the statistical modelling.

## 3. Data Description

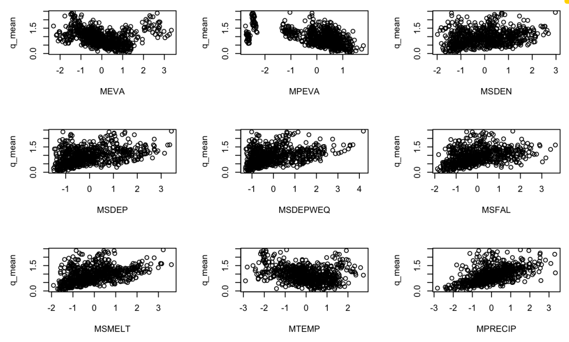
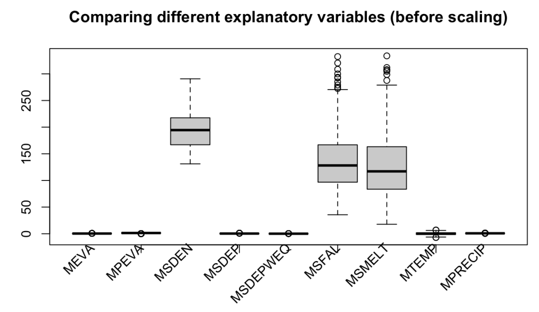
The data was collected daily by satellites and an aggregation of the data (annual averages) was provided for the propose of this analysis. Data contains observations from 23 medium-sized water catchment areas located around Canada. The size of these watershed areas ranges from 50 $km^2$ to 10,000 $km^2$. The data of various climate variables was taken from the year 1980 to 2018. There are 774 rows and 12 columns with no missing data. An additional dataset with the watershed shape data (Longitude, Latitude) was also provided by the client for further analysis of the effect of spatial features on the streamflow. *Please refer to the Appendix I (A.1.0) for the full data dictionary (w/ abbrev.)*

Table 1: Description of variables used for analysis

| | **Variable** | **Unit** | **Description** |
|---|---|---|---|
| 1. | Mean Yearly Evaporation | $m$ | Depth of water evaporates from the catchment area |
| 2. | Mean Yearly Potential Evaporation | $ml$ | The amount of evaporation that would occur if a sufficient water source were available |
| 3. | Mean Snow Density | $kg^3m^{-3}$ | The density of the snow |
| 4. | Mean Snow Depth | $m$ | The depth of new and old snow that remains on the ground at observation time |
| 5. | Mean Snow Depth of Snow Water Equivalent | $m$ | Water equivalent of melted snow collected in the gauge since the last observation |
| 6. | Mean Yearly Temperature | ř$C$ | The temperature measured in °C |
| 7. | Mean Yearly Snowfall | $m$ | The record of snowfall (snow, ice pellets) since the previous snowfall observation (24 hours) |
| 8. | Mean Yearly Snowmelt | $m$ | The depth of runoff produced from melting snow |
| 9. | Mean Yearly Total Precipitation | $m$ | The depth of total rainfall and water-equivalent of snowfall |
| 10. | Year | year | The year the data was recorded in |
| 11. | Grid Code | - | Grid code where the watershed is located |
| 12. | Annual Average Streamflow (Response) | $m^3\ year^{-1}$ | The average daily streamflow recorded for a year |

Table 2: Summary Statistics of all the variables

| **Var** | **EVA** | **PEVA** | **SDEN** | **SDEP** | **SWEQ** | **SFAL** | **SMELT** | **TEMP** | **PREC** | **Q** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Min** | 0.123 | 0.072 | 131.1 | 0.048 | 0.008 | 35.61 | 17.78 | -7.153 | 0.364 | 0.045 |
| **25%** | 0.293 | 1.141 | 166.9 | 0.213 | 0.044 | 96.80 | 83.57 | -1.417 | 0.684 | 0.599 |
| **50%** | 0.352 | 1.353 | 194.5 | 0.319 | 0.077 | 128.1 | 117.0 | 0.190 | 0.768 | 0.899 |
| **Mean** | 0.364 | 1.238 | 195.0 | 0.376 | 0.097 | 136.6 | 127.2 | 0.135 | 0.779 | 0.949 |
| **75%** | 0.427 | 1.509 | 217.5 | 0.507 | 0.133 | 166.8 | 163.3 | 1.806 | 0.872 | 1.233 |
| **Max** | 0.7112 | 1.999 | 290.7 | 1.120 | 0.382 | 332.5 | 333.8 | 7.014 | 1.270 | 2.436 |

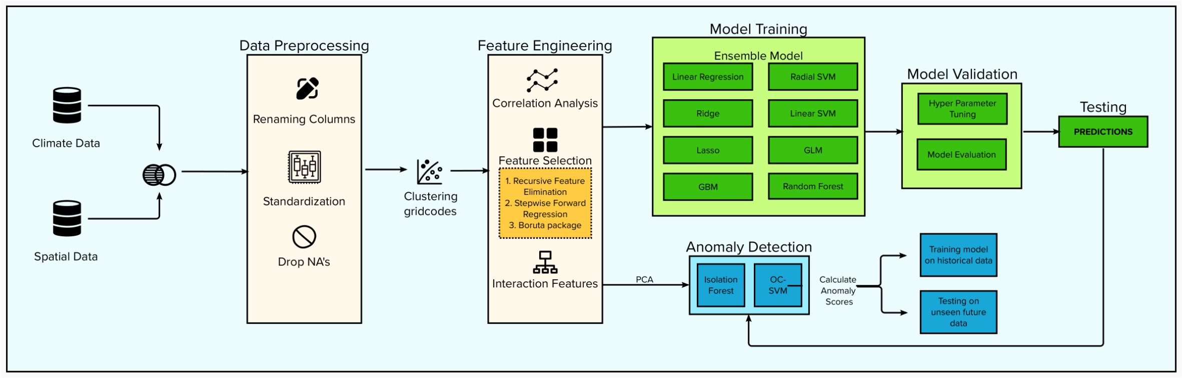
| Fig 1: Relationship b/w explanatory & response variables | Fig 2: Comparing different explanatory variables (before scaling) |
|---|---|
|  |  |

## 4. Methods

### 4.0. Pipeline

Below in Fig 3, we have a Proof-of-Concept pipeline that addresses all of the ~~client's~~ research questions. A breakdown of each of the steps shown in the end-to-end workflow diagram is covered below. We have successfully completed our Model Training and Initial Testing. Our future work will encompass performing Hyperparameter Tuning and implementing the Anomaly Detection system.

Fig 3: End-to-End Pipeline



### 4.1. K-Means and Hierarchical Clustering (by gridcodes)

From Fig 4 and Fig 5, we see that the `gridcode` variable is useful. In our initial analysis, we used the cluster variable generated from this clustering as one of our features but it was not included in our final feature space due to redundancy. We can either use gridcode or replace it with a combination of features: cluster, Latitude and Longitude. However, while testing on unseen data with new gridcodes, using the spatial features will be a more effective solution to make our predictions.

Fig 4: K-Means Clustering

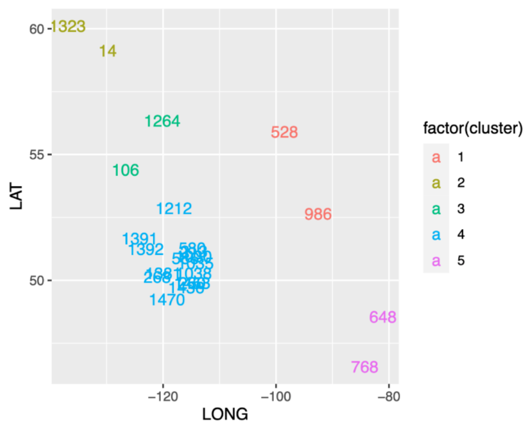
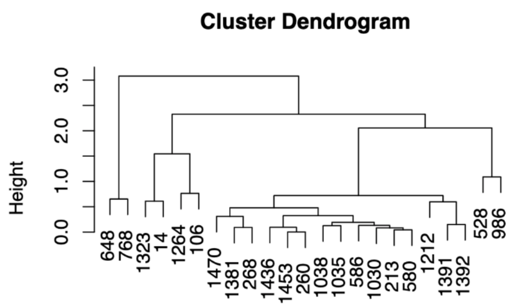


Fig 5: Hierarchical Clustering



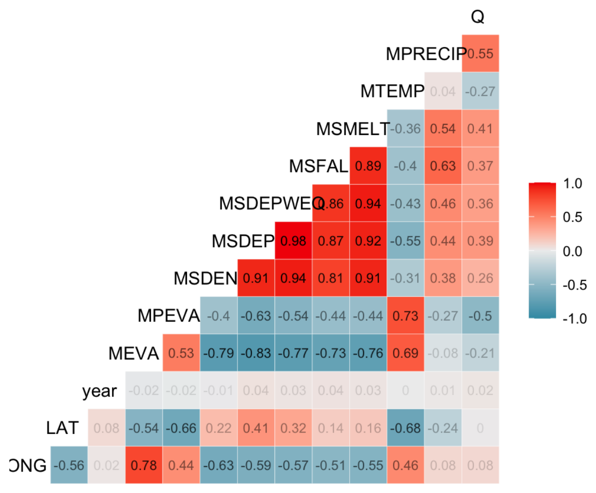
### 4.2. Feature Selection

One of the most important step in our pipeline was Feature Selection and we have used several different methods to do so:

#### 4.2.1. Pairwise Correlation Analysis

Before using any of the traditional feature selection techniques mentioned below, we investigated if any of the features were highly correlated. We found out that the Mean Snow Depth, Mean Snow Depth of Snowwater Equivalent and the Mean Yearly Snowmelt were highly correlated. For our initial ~~MVP, we have dropped these features as there is insufficient information contained in the linear combination of the above features leading to redundancy.~~ However, as part of our future work, we can perform PCA as well to deal with this issue.

Fig 6: Heatmap presenting correlation between variables



### 4.2.2 Exhaustive Search using Regsubsets()

After excluding the highly correlated variables, we used this method to find the most important variables: Mean Yearly Total Precipitation (MPRECIP), Mean Yearly Temperature (MTEMP), Mean Yearly Potential Evaporation (MPEVA), Mean Yearly Evaporation (MEVA), Mean Snow Density (MSDEN), Mean Yearly Snowfall (MSFAL)

### 4.2.2. Recursive Feature Elimination using Random Forest

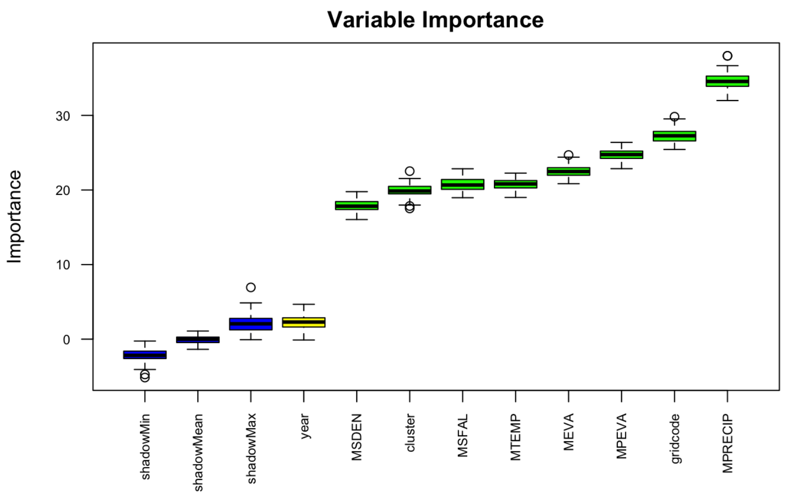
We used RFE to get the most important features that will contribute to the predictive performance of out models. The important features included:gridcode, ~~Mean Yearly Total Precipitation~~ (MPRECIP), ~~Mean Yearly Temperature~~ (MTEMP), ~~Mean Yearly Potential Evaporation~~ (MPEVA), ~~Mean Yearly Evaporation~~ (MEVA), ~~Mean Snow Density~~ (MSDEN), ~~Mean Yearly Snowfall~~ (MSFAL)

### 4.2.3. Variable Importance using `boruta` package

From Fig 7, we see that the most important variables selected ranked in the following order: ~~Mean Yearly Total Precipitation (MPRECIP), gridcode, Mean Yearly Potential Evaporation (MPEVA), Mean Yearly Evaporation (MEVA), Mean Yearly Temperature (MTEMP), Mean Yearly Snowfall (MSFAL), cluster, Mean Snow Density (MSDEN)~~

It is to be noted that the results from the above methods used for feature selection align with each other ~~and are hence reliable~~.

Fig 7: Variable Importance (using 'boruta' package)



### 4.2.4. Forward Stepwise Regression

This method was used to find important variables as well but it was exclusively used to find the important interaction *(please refer to A.1.1 for the definition)* terms for the Linear models. The two plots below are comparing the performance metrics of the linear model with different subset sizes of the feature space (with/without interaction terms respectively). We see that after 6 features without any interaction terms, the curve for all metrics plateaus indicating that 6 features is a good subset size to maximize the performance. Similarly, when including the interaction terms, we see the same result after 19 variables. *(The complete list of variables can be found in the Appendix I (A.1.2)).*

Fig 8: Feature Selection w/o Interaction terms (Forward Stepwise Regression)

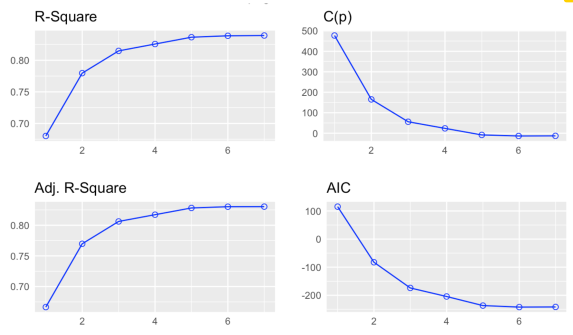
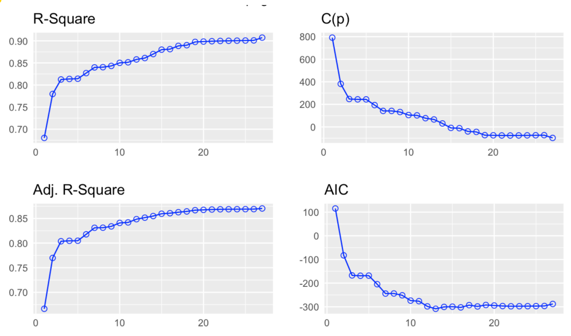


Fig 9: Feature Selection w/ Interaction terms (Forward Stepwise Regression)



### 4.3. Model Training

A ~~good~~ variety of models were implemented as a part of our analysis ~~for extensive results~~. Two sets of models where trained to capture both the effect of individual co-variate terms and the interaction terms on the predictive performance of the model. Please refer to Table 3 and Table 4 for ~~results that denote~~ the predictive performance of the models.

#### 4.3.1. Linear Models

We start with Linear Regression as our baseline model. After applying the above feature selection methods, we train a Generalized Linear Model and some Regularized Linear Models like the Lasso and Ridge Regression with a 5-fold cross validation to test the model performance on the training data. We train a similar set of models with interaction terms as features and we see slightly better results than the former method.

#### 4.3.2. Support Vector Machines

To add further complexities to the previous models, we trained Support Vector Machines that expands our feature space using different kernels. We have used a linear and a radial kernel to compare performance of models with and without the complexities relating to Linearity respectively. In the latter kernel, only the neighboring behavior of data is taken into account which means only those data points influence the modelling compared to to former whose performance is similar to a Linear model. From Table 3 and Table 4, we see that the Linear SVM with interaction terms performs slightly better than Linear SVM without the interaction terms. ~~Wheras,~~ the Radial SVM without interaction terms performs slightly better than Radial SVM with the interaction terms.

#### 4.3.3. Tree Models

We used two tree-based models such as Random Forest and Gradient Boosting Machine to improve the performance compared to the above models. From Table 3 and Table 4, we see that the Random Forest with interaction terms has comparable results with Random Forest without interaction terms. However, the GBM without interaction terms gives surprising better results than the GBM with interaction terms.

### 4.3.3. Ensemble Models

We trained two Ensemble models with/without the interaction terms that combine the above listed 8 models to produce improved results. These models generally produce more accurate predictions than a single model. From Table 3 and Table 4, we see that that the Ensemble Model with interaction terms has significantly better results compared to the Ensemble Model without interaction terms. Therefore, based on the testing evaluation metrics from Table 3 and 4, we chose the Ensemble Model with the interaction terms as our best model.

Fig 10: Comparing In-sample prediction performance for different models w/o Interaction terms

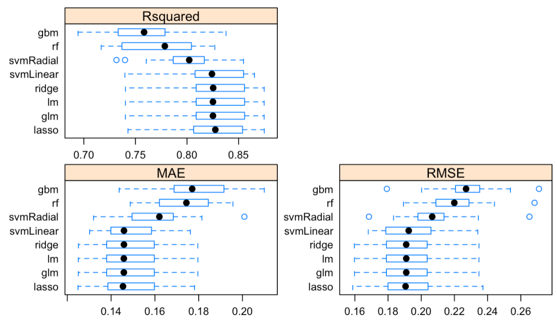
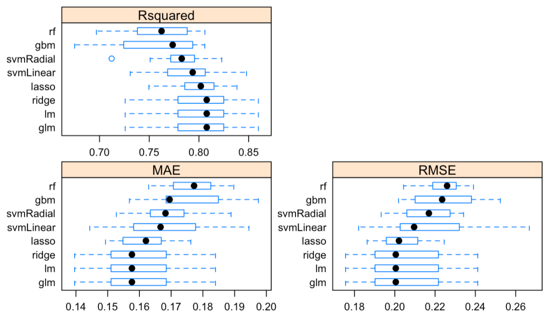


Fig 11: Comparing In-sample prediction performance for different models w/ Interaction terms



## 5. Results

From Table 3, we see that the GLM and the Linear Regression model have the lowest RMSE and MAE with the highest R2 value, meaning they have the best predictive performance out of all the models with no interaction terms.

Table 3: Comparing out-of-sample evaluation metrics of different model without the interaction terms

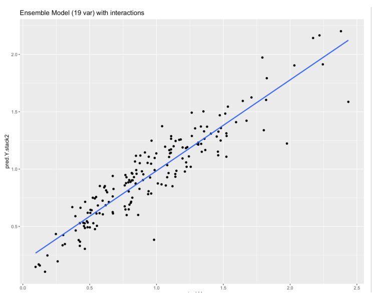
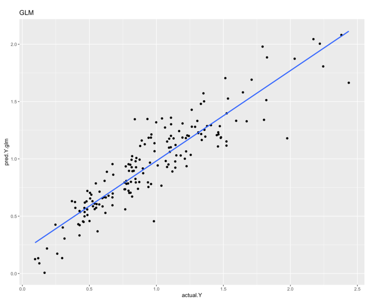
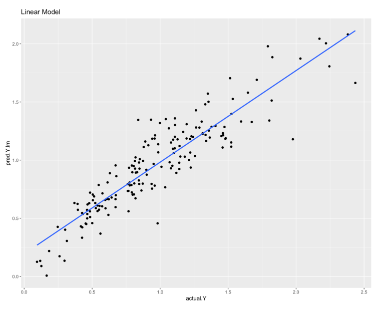
| **Models** | **RMSE** | **R2** | **MAE** | |
|---|---|---|---|---|
| Random Forest | 0.2479 | 0.7123 | 0.1884 | |
| Gradient Boosting Machine | 0.2401 | 0.7301 | 0.1847 | |
| Linear Regression | 0.1928 | 0.8260 | 0.1447 | (*) |
| Lasso Regression | 0.1940 | 0.8239 | 0.1455 | |
| Ridge Regression | 0.1928 | 0.8259 | 0.1448 | |
| Generalized Linear Model | 0.1928 | 0.8260 | 0.1447 | (*) |
| Linear Support Vector Machine | 0.1966 | 0.8191 | 0.1479 | |
| Radial Support Vector Machine | 0.2122 | 0.7892 | 0.1574 | |
| Ensemble Model | 0.1959 | 0.8204 | 0.1474 | |

From Table 4, we see that once again that the ensemble of all the 8 models listed has the lowest RMSE and MAE with the highest R2 value, meaning it has the best predictive performance out of all the individual models with interaction terms.

Table 4: Comparing out-of-sample evaluation metrics of different model with the interaction terms

| **Models** | **RMSE** | **R2** | **MAE** | |
|---|---|---|---|---|
| Random Forest | 0.2442 | 0.7208 | 0.1857 | |
| Gradient Boosting Machine | 0.2272 | 0.7584 | 0.1679 | |
| Linear Regression | 0.1854 | 0.8392 | 0.1416 | |
| Lasso Regression | 0.1918 | 0.8278 | 0.1485 | |
| Ridge Regression | 0.1926 | 0.8263 | 0.1452 | |
| Generalized Linear Model | 0.1854 | 0.8392 | 0.1416 | |
| Linear Support Vector Machine | 0.1886 | 0.8335 | 0.1445 | |
| Radial Support Vector Machine | 0.2233 | 0.7665 | 0.1634 | |
| Ensemble Model | 0.1795 | 0.8492 | 0.1359 | (*) |

The plots below show how much the prediction from our best 3 models deviated from actual value therefore giving us a rough estimation of whether a model is a good fit or not. We clearly see that the latter two plots for GLM and Linear Regression are almost comparable whereas for the Ensemble Model, the points are closer to the fitted line indicating a ~~good~~ fit.

| Fig 12: Predicted vs Actual Values (Ensemble Model) | Fig 13: Predicted vs Actual Values (GLM) | Fig 11: Predicted vs Actual Values (Linear Regression) |
|---|---|---|
|  |  |  |

Comparing the predictive performance of the best models from both Table 3 and Table 4, we see that the latter (Ensemble Model, RMSE: 0.1795) has a better performance. Therefore, the best model we chose for making predictions was the Ensemble Model with the interaction terms.

## 6. Conclusion

After all the model iterations and improvements, we were able to achieve ~~fairly~~ good results with the Ensemble Model taking into account the interaction effect between variables. These results have large implications when it comes to water resource management both financially and economically. We have conducted our primary analysis taking into account the spatial data (e.g. gridcodes) which serves as a good MVP to predict the streamflow.

As a side interest and to expand upon the idea of predicting the streamflow solely using climate variables and not any spatial and temporal data, we conducted an analysis and trained models without this data and the predictive performance dropped significantly. Although this addresses the ~~client's~~ first research question of whether one catchment can be used to extrapolate stream flow in another catchment, the limitation we faced was lack of training data. To be able to model such a complex problem using climatic variables, we need more data to train our model which can help with improving the predictive performance of the model.

To address the second research question of whether or not we can detect the unusual streamflow activity accurately, we built a proof of concept pipeline for the outlier detection system. It will take the features from our prediction model as input and label the observations as either anomalous or regular depending on the anomaly scores which are the measures of deviation from normal behavior. We will face the same challenge here - lack of training data~~,~~. This will lead to increase in false positives and false negatives in the outlier detection system which can have detrimental consequences. For example, not being able to detect a subtle increase in the streamflow (false negatives) which could lead to irrigation problems and in severe cases even floods. Or getting a huge pool of outlier values (false positives) that will raise false alarms of anomalous behaviour more often than desired.

### 6.1. Future Steps

After modelling the data with the techniques above, we believe the results could be further improved by investigating some aspects further more and in order to deliver the complete pipeline, we will need focus on the follwoing next steps:

- Perform Hyperparameter Tuning to fine-tune the model and carry out model validation
- Find better techniques to select the best interaction variables and comparing its results with the Forward Stepwise Regression
- Implement the outlier detection system from he pipeline using Isolation Forest, OC-SVM models and testing it on unseen data.

## 7. References

- Government of Canada / Gouvernement du Canada. (2021, November 25). Government of Canada / gouvernement du Canada. Climate. Retrieved February 5, 2022, from https://climate.weather.gc.ca/glossary_e.html
- US Department of Commerce, N. O. A. A. (2012, March 8). Snow measurement guidelines. Snow Measurement Guidelines. Retrieved February 5, 2022, from https://www.weather.gov/gsp/snow
- Janssen, J., & Ameli, A. A. (2021). A Hydrologic Functional Approach for Improving Large-Sample Hydrology Performance in Poorly Gauged Regions. Water Resources Research, 57(9), e2021WR030263.

## 8. Appendix I (For Client)

### A.1.0 Data Dictionary

| **Variable** | **Name** |
|---|---|
| MPRECIP | Mean Yearly Total Precipitation |
| MPEVA | Mean Yearly Potential Evaporation |
| MEVA | Mean Yearly Evaporation |
| MSFAL | Mean Yearly Snowfall |
| MSDEN | Mean Snow Density |
| MTEMP | Mean Yearly Temperature |
| MSMELT | Mean Yearly Snowmelt |
| MSDEP | Mean Snow Depth |
| MSDEPWEQ | Mean Snow Depth of Snowwater Equivalent |

### A.1.1 Interaction Term

> "In statistics, an interaction is a special property of three or more variables, where two or more variables interact to affect a third variable in a non-additive manner. In other words, the two variables interact to have an effect that is more than the sum of their parts."

**Reference**: Statistical interaction: More than the sum of its parts. Statistics Solutions. (2021, June 22). Retrieved February 21, 2022, from https://www.statisticssolutions.com/statistical-interaction-more-than-the-sum-of-its-parts/

### A.1.2. Important Variables using Forward Stepwise Regression (including interaction terms)

| | |
|---|---|
| gridcode | MEVA:MSDEN |
| year:MPRECIP | MEVA:MSFAL |
| MEVA | grid-code:MEVA |
| MTEMP | gridcode:MTEMP |
| year | MPEVA:MTEMP |
| MSDEN | gridcode:MPRECIP |
| MPEVA | MSFAL:MPRECIP |
| MSFAL:MTEMP | gridcode:MSDEN |
| MTEMP:MPRECIP | MEVA:MPEVA |
| MPRECIP | |

## 9. Appendix II (For Mentor)

### 9.1. Data Examination

```
# Sys.setenv(LANG = "en")
# library(tidyverse)
# library(ggplot2)
# library(GGally)
# library(olsrr)
# library(gridExtra)
# library(cluster)
# library(factoextra)
# library(caretEnsemble)
# library(caret)
# library(mlbench)
# library(Metrics)
# library(Boruta)
```

```
### Change column names for convenience

# dt <- read_csv('data_stat450.csv')
# dt <- rename(dt, MEVA = 'Mean Evaporation (m_per_year)',
#              MPEVA = 'Mean Potential Evaporation (m_per_year)',
#              MSDEN = 'Mean Snow Density (kg3_per_m3)',
#              MSDEP = "Mean Snow Depth (m)",
#              MSDEPWEQ = "Mean Snow Depth, Snow Water Equiv (m_of_swe)",
#              MSFAL = "Mean Snowfall (m_per_year)",
#              MSMELT = "Mean Snowmelt (m_per_year)",
#              MTEMP = "Mean Temperature (deg_C)",
#              MPRECIP = "Mean Total Precip (m_per_year)",
#              Q = "q_mean") |>
#   mutate(gridcode = as.factor(gridcode),
#          year = as.integer(year),
#          MPEVA = MPEVA/1000)
#
# grid <- read_csv('grid_codes.csv') |>
#   select(gridcode, Longitude, Latitude) |>
#   mutate(gridcode = as.factor(gridcode))
#
# dt <- left_join(grid,dt)
# dt <- rename(dt, LONG = 'Longitude', LAT = 'Latitude')
# head(dt)
# colnames(dt)
# length(unique(dt$gridcode))
```

### 9.2. Explanatory Data Analysis

```
### Check dimensions and NAs
# summary(dt)
# nrow(dt)
# ncol(dt)
# any(is.na(dt[1:13]))
# any(is.na(dt['Q']))
# sum(is.na(dt['Q']))
# # dt <- dt %>%drop_na()
```

14 columns, 774 rows Only the last column (response var) have 61 missing values.

```
# dt %>% group_by(gridcode) %>% summarise(n=n()) |>
#   ggplot(aes(x=n)) + geom_bar()
```

There are missing years in most of the locations (grid codes): the 'n' column have very different values.

### 9.2.1. Distribution & Standardization

```
# par(mar=c(6, 4.1, 4.1, 2.1))
# labels <- paste(colnames(dt[,5:13]))
# boxplot(dt[,5:13],  xaxt = "n",xlab = "",
#         main = 'Comparing different explanatory variables (before scaling)')
# axis(1, labels = FALSE)
# text(x =  seq_along(labels), y = par("usr")[3] - 1, srt = 45, adj = 1,
#      labels = labels, xpd = TRUE)
```

```
# dt[,5:13] = scale(dt[,5:13])
# scaled_dt = dt
#
# head(scaled_dt)
#
# par(mar=c(6, 4.1, 4.1, 2.1)+4)
# labels <- paste(colnames(scaled_dt[,5:13]))
# boxplot(scaled_dt[,5:13],  xaxt = "n",xlab = "",
#         main= 'Comparing different explanatory variables (after scaling)')
# axis(1, labels = FALSE)
# text(x =  seq_along(labels), y = par("usr")[3] - 1, srt = 60, adj = 1,
#      labels = labels, xpd = TRUE)
```

```
# summary(lm(Q ~ .-gridcode-year-LONG-LAT, data = scaled_dt))$adj.r.squared
# summary(lm(Q ~ .-gridcode-year-LONG-LAT, data = dt))$adj.r.squared
# # same results regardless of the scaling
```

### 9.2.2. Relationship b/w explanatory & response variables

```
# par(mfrow= c(3,3))
# for (i in colnames(scaled_dt[,5:13])) {
#
#   x <- pull(scaled_dt, 'Q')
#
#   # Plot histogram of x
#   plot(pull(scaled_dt, i), x, ylab = "q_mean", xlab = i)
#
# }
```

#### 9.2.3. K-Means Clustering & Hierarchical Clustering (grid code)

#### 9.2.4. Box plot for climate variables

### 9.2.5. Correlation & Heatmap

```
#ggpairs(dt,lower = list(continuous = wrap("smooth", alpha = 0.1, size=0.05)),
#        upper = list(continuous = wrap("cor", size=2)))
# ggcorr(dt[0:15], label = TRUE,
#        hjust =0.7, size = 4, label_size = 3, label_round = 2, label_alpha = TRUE)
```

### 9.2.6. Time series effect

```
# for (i in unique(dt$gridcode)) {
#   temp <- dt |> filter(gridcode==i)
#   T <- ts(temp['MEVA'],start=c(1981), frequency = 1)
#   par(mfrow=c(2,1))
#   acf(T, xlab = i)
#   pacf(T, xlab = i)
# }
```

Mostly no significant annual trend / time series effect.

## 9.3. Modeling

#### 9.3.1. Data Spliting

```
# set.seed(123)
# dt.valid <- na.omit(dt)
# inTrain <- createDataPartition(y=dt.valid$Q, p=.75,list=F)
# training <- dt.valid[inTrain,]
# nrow(training)
# testing <- dt.valid[-inTrain,]
# nrow(testing)
```

#### 9.3.2 Feature Selection

```

# set.seed(7)
# # calculate correlation matrix
# correlationMatrix <- cor(dt[,6:14])
# # summarize the correlation matrix
#
# # find attributes that are highly corrected (ideally >0.75)
# highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.9,
#                                      exact = TRUE, names = TRUE,verbose = TRUE)
# # print indexes of highly correlated attributes
# print(highlyCorrelated)
#
# training = training |> select(-highlyCorrelated)
# testing = testing |> select(-highlyCorrelated)
```

#### 9.3.2.1. Pairwise Correlation

```
# set.seed(7)
#
# # define the control using a random forest selection function
# control <- rfeControl(functions=rfFuncs, method="cv", number=10)
# # run the RFE algorithm
# results <- rfe(training[,5:11], as.matrix(training[,12]), rfeControl=control)
# # summarize the results
# print(results)
# # list the chosen features
# predictors(results)
# # plot the results
# plot(results, type=c("g", "o"), main="RMSE vs # variables")
```

#### 9.3.2.2. Recursive Feature Selection through Random Forest

```
# training = training |> select(-c('LONG', 'LAT'))
# boruta_output <- Boruta(Q ~ ., data=na.omit(training), doTrace=0)
# roughFixMod <- TentativeRoughFix(boruta_output)
# boruta_signif <- getSelectedAttributes(roughFixMod)
#
# # Variable Importance Scores
# imps <- attStats(roughFixMod)
# imps2 = imps[imps$decision != 'Rejected', c('meanImp', 'decision')]
# head(imps2[order(-imps2$meanImp), ])  # descending sort
#
# # Plot variable importance
# plot(boruta_output, cex.axis=.7, las=2, xlab="", main="Variable Importance")

## Notes
## regsubsets: gridcode MEVA MSDEN MSFAL MTEMP MPRECIP
## Excluding highly correlated features:
## "MSDEP"    "MSDEPWEQ" "MSMELT"
## Matching results from previous method:
## "MPRECIP"  "MTEMP"    "MPEVA"  "MEVA" "MSDEN"  "MSFAL"
## Results from this method that coincide with previous:
## 'MPRECIP' "MPEVA" "gridcode" "MTEMP" "MEVA" "MSFAL" "MSDEN"
## Exclude LAT and LONG
```

#### 9.3.2.3.Feature Selection using `boruta`

### 9.3.3. Linear Regression (NO INTERACTION)

```
# set.seed(10567)
#
# M0 <-lm(Q ~ .-year-cluster-LONG-LAT, data = training)
#
# M0.1 <- train(Q ~ .-year-cluster-LONG-LAT, data = training, method = "lm")
# summary(M0.1)$adj.r.squared
# M0.1$results
```

**Model v1.0 baseline**

**Model v1.0: Feature subset selection through regsubsets()**

```
# set.seed(1098)
# M1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode + MTEMP +
#                  MEVA + MSFAL + MSDEN, data = training, method = "lm")
# M1.1$results
```

**Model v1.1 with best 7 variables**

### 9.3.4. Linear Regression (with INTERACTION)

```
# set.seed(42)
# dt_temp <- training |> select(-cluster)
# M2<-lm(Q ~ (.)^2, data = dt_temp)
# # fitted # of terms
# length(M2$coefficients)-1
# # adj. R^2
# summary(M2)$adj.r.squared
```

**Model v2.0 baseline**

```
# OLS2 <- ols_step_forward_p(M2)
# OLS2
# plot(OLS2)
```

**Model v2.0: Variable selection (using forward stepwise)**

```
# set.seed(123)
# M2.2 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+
#                 MSDEN + MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+
#                 gridcode:MTEMP + MPEVA:MTEMP + gridcode:MPRECIP +
#                 MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#               data = training, method = "lm")
# M2.2$results
```

**Model v2.1 with best 19 variables (including interactions)**

## 9.5. Model Comparison

### 9.5.1. Models with 7 variables

```
# set.seed(7452)
# trControl <- trainControl(method = "repeatedcv",
#     savePredictions = 'final',
#     number = 5,
#     repeats = 3,
#     search = "grid")
#
# algo.list = c('rf', 'gbm', 'lm', 'lasso','ridge','glm', 'svmLinear','svmRadial')
# models1 <- caretList(Q ~ MPRECIP + MPEVA + gridcode + MTEMP +
#                         MEVA + MSFAL + MSDEN, data = training,
#                       methodList = algo.list, trControl = trControl)
# results1 <- resamples(models1)
#
# summary(results1)
# stack1 = caretStack(models1, method="glm", metric="RMSE", trControl=trControl)
# stack1
```

```
# scales1 = list(x=list(relation='free'), y=list(relation='free'))
# bwplot(results1,scales = scales1,layout = c(2,2))
```

### 9.5.2. Models with best 19 variables (including interactions)

```
# set.seed(45)
# algo.list = c('rf', 'gbm', 'lm', 'lasso','ridge','glm', 'svmLinear','svmRadial')
# models2 <- caretList(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+
#                        year+ MSDEN + MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                     MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+
#                     gridcode:MTEMP + MPEVA:MTEMP + gridcode:MPRECIP +
#                   MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                   data = training, methodList = algo.list, trControl = trControl)
# results2 <- resamples(models2)
# summary(results2)
#
# stack2 = caretStack(models2, method="glm", metric="RMSE", trControl=trControl)
# stack2
```

```
# scales2 = list(x=list(relation='free'), y=list(relation='free'))
# bwplot(results2,scales = scales2,layout = c(2,2))
```

## 9.6. Prediction (with testing dataset)

### 9.6.1. Comparing Models with 7 variables

```
# set.seed(452)
# lm1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode + MTEMP +
#                  MEVA + MSFAL + MSDEN, data = training, method = "lm")
# glm1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode + MTEMP +
#                   MEVA + MSFAL + MSDEN, data = training, method = "glm")
# ridge1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode + MTEMP +
#                     MEVA + MSFAL + MSDEN, data = training, method = "ridge")
# lasso1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode +
#                     MTEMP + MEVA + MSFAL + MSDEN, data = training, method = "lasso")
# rf1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode +
#                     MTEMP + MEVA + MSFAL + MSDEN, data = training, method = "rf")
# gbm1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode +
#                     MTEMP + MEVA + MSFAL + MSDEN, data = training, method = "gbm")
# svml1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode +
#                     MTEMP + MEVA + MSFAL + MSDEN, data = training, method = "svmLinear")
# svmr1.1 <- train(Q ~ MPRECIP + MPEVA + gridcode +
#                     MTEMP + MEVA + MSFAL + MSDEN, data = training, method = "svmRadial")
# actual.Y <- testing$Q
# pred.Y.lm <- predict(lm1.1, newdata = testing)
# pred.Y.glm <- predict(glm1.1, newdata = testing)
# pred.Y.ridge <- predict(ridge1.1, newdata = testing)
# pred.Y.lasso <- predict(lasso1.1, newdata = testing)
# pred.Y.rf <- predict(rf1.1, newdata = testing)
# pred.Y.gbm <- predict(gbm1.1, newdata = testing)
# pred.Y.svml <- predict(svml1.1, newdata = testing)
# pred.Y.svmr <- predict(svmr1.1, newdata = testing)
# pred.Y.stack1 <- predict(stack1, newdata = testing)
#
# ggplot(,aes(x=actual.Y, y=pred.Y.lm)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Linear Model')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.glm)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('GLM')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.ridge)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Ridge')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.lasso)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Lasso')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.stack1)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Ensemble Model (7 var)')
#
# eval_results <- function(true, predicted) {
#   SSE <- sum((predicted - true)^2)
#   SST <- sum((true - mean(true))^2)
#   R2 <- 1 - SSE / SST
#   return(R2)}
#
# print("LM")
# rmse(actual.Y, pred.Y.lm)
# mae(actual.Y, pred.Y.lm)
# eval_results(actual.Y, pred.Y.lm)
#
# print("GLM")
# rmse(actual.Y, pred.Y.glm)
# mae(actual.Y, pred.Y.glm)
# eval_results(actual.Y, pred.Y.glm)
#
# print("RIDGE")
# rmse(actual.Y, pred.Y.ridge)
# mae(actual.Y, pred.Y.ridge)
# eval_results(actual.Y, pred.Y.ridge)
#
# print("LASSO")
# rmse(actual.Y, pred.Y.lasso)
# mae(actual.Y, pred.Y.lasso)
# eval_results(actual.Y, pred.Y.lasso)
#
# print("RF")
# rmse(actual.Y, pred.Y.rf)
# mae(actual.Y, pred.Y.rf)
# eval_results(actual.Y, pred.Y.rf)
```

```
#
# print("GBM")
# rmse(actual.Y, pred.Y.gbm)
# mae(actual.Y, pred.Y.gbm)
# eval_results(actual.Y, pred.Y.gbm)
#
# print("SVML")
# rmse(actual.Y, pred.Y.svml)
# mae(actual.Y, pred.Y.svml)
# eval_results(actual.Y, pred.Y.svml)
#
# print("SVMR")
# rmse(actual.Y, pred.Y.svmr)
# mae(actual.Y, pred.Y.svmr)
# eval_results(actual.Y, pred.Y.svmr)
#
# print("ENSEMBLE")
# rmse(actual.Y, pred.Y.stack1)
# mae(actual.Y, pred.Y.stack1)
# eval_results(actual.Y, pred.Y.stack1)
```

### 9.6.2. Comparing Models with 19 variables (including interactions)

```
# set.seed(300)
# lm2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                  MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                  MPEVA:MTEMP + gridcode:MPRECIP +
#                 MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                data = training, method = "lm")
# glm2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                   MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                   MPEVA:MTEMP + gridcode:MPRECIP +
#                 MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                 data = training, method = "glm")
# ridge2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                     MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                   MPEVA:MTEMP + gridcode:MPRECIP +
#                 MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                 data = training, method = "ridge")
# lasso2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                     MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                   MPEVA:MTEMP + gridcode:MPRECIP +
#                 MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                 data = training, method = "lasso")
#
# rf2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                     MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                 MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                    MPEVA:MTEMP + gridcode:MPRECIP +
#                  MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                  data = training, method = "rf")
#
# gbm2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                      MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                  MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                    MPEVA:MTEMP + gridcode:MPRECIP +
#                  MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                  data = training, method = "gbm")
#
# svml2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                      MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                  MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                    MPEVA:MTEMP + gridcode:MPRECIP +
#                  MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                  data = training, method = "svmLinear")
#
# svmr2.1 <- train(Q ~ gridcode + year:MPRECIP + MEVA +  MTEMP+ year+ MSDEN +
#                      MPEVA+ MSFAL:MTEMP + MTEMP:MPRECIP + MPRECIP +
#                  MEVA:MSDEN + MEVA:MSFAL + gridcode:MEVA+ gridcode:MTEMP +
#                    MPEVA:MTEMP + gridcode:MPRECIP +
#                  MSFAL:MPRECIP + gridcode:MSDEN + MEVA:MPEVA,
#                  data = training, method = "svmRadial")
#
# actual.Y <- testing$Q
# pred.Y.lm <- predict(lm2.1, newdata = testing)
# pred.Y.glm <- predict(glm2.1, newdata = testing)
# pred.Y.ridge <- predict(ridge2.1, newdata = testing)
# pred.Y.lasso <- predict(lasso2.1, newdata = testing)
# pred.Y.rf <- predict(rf2.1, newdata = testing)
# pred.Y.gbm <- predict(gbm2.1, newdata = testing)
# pred.Y.svml <- predict(svml2.1, newdata = testing)
# pred.Y.svmr <- predict(svmr2.1, newdata = testing)
# pred.Y.stack2 <- predict(stack2, newdata = testing)
#
# ggplot(,aes(x=actual.Y, y=pred.Y.lm)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Linear Model')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.glm)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('GLM')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.ridge)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Ridge')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.lasso)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Lasso')
#
# ggplot(,aes(x=actual.Y, y=pred.Y.stack2)) +
#   geom_point() +
#   geom_smooth(method = "lm", se = FALSE) +
#   ggtitle('Ensemble Model (19 var) with interactions')
#
# eval_results <- function(true, predicted) {
#   SSE <- sum((predicted - true)^2)
#   SST <- sum((true - mean(true))^2)
#   R2 <- 1 - SSE / SST
#   return(R2)}
#
# print("LM")
# rmse(actual.Y, pred.Y.lm)
# mae(actual.Y, pred.Y.lm)
# eval_results(actual.Y, pred.Y.lm)
#
# print("GLM")
# rmse(actual.Y, pred.Y.glm)
# mae(actual.Y, pred.Y.glm)
# eval_results(actual.Y, pred.Y.glm)
#
# print("RIDGE")
# rmse(actual.Y, pred.Y.ridge)
# mae(actual.Y, pred.Y.ridge)
# eval_results(actual.Y, pred.Y.ridge)
#
# print("LASSO")
# rmse(actual.Y, pred.Y.lasso)
# mae(actual.Y, pred.Y.lasso)
# eval_results(actual.Y, pred.Y.lasso)
#
# print("RF")
# rmse(actual.Y, pred.Y.rf)
# mae(actual.Y, pred.Y.rf)
# eval_results(actual.Y, pred.Y.rf)
#
# print("GBM")
# rmse(actual.Y, pred.Y.gbm)
# mae(actual.Y, pred.Y.gbm)
# eval_results(actual.Y, pred.Y.gbm)
#
# print("SVML")
# rmse(actual.Y, pred.Y.svml)
# mae(actual.Y, pred.Y.svml)
# eval_results(actual.Y, pred.Y.svml)
#
# print("SVMR")
# rmse(actual.Y, pred.Y.svmr)
# mae(actual.Y, pred.Y.svmr)
# eval_results(actual.Y, pred.Y.svmr)
```

```r
#
# print("ENSEMBLE")
# rmse(actual.Y, pred.Y.stack2)
# mae(actual.Y, pred.Y.stack2)
# eval_results(actual.Y, pred.Y.stack2)
```