# **California Ground Water Dataset : Prediction of Groundwater depth**

NOTE: GSE\_GWE is term that will refer to the depth of ground water. GSE is ground surface elevation and GWE is ground water elevation. The difference between the two is the comparable groundwater depth for all wells.

INTRODUCTION

{Project Proposal}

TIMESERIES ANALYSIS AS A SUPERVISED LEARNING TASK

Univariate timeseries can be modeled using auto-regression, moving averages and a combination of these using ARIMA, SARIMA and other forecasting techniques. These require studying the series for stationarity (**the properties – mean/trend, variance and covariance, do not vary with time**), and seasonality to determine the best algorithm to pick. There is also a concept of hierarchical time series where an existing structure of hierarchy of say continents, countries, states constitute a time series granularity structure. Forecasts for such hierarchical time series should be coherent, meaning that the forecast for an upper-level time series equals the sum of forecasts for corresponding lower-level time series.

Multivariate timeseries without groups can be analyzed using Vector Auto Regression technique. In a VAR model, each variable is a linear function of the past values of itself and the past values of all the other variables.  Each variable is using the past values of every variable to make the predictions. Unlike Auto Regression, **VAR is able to understand and use the relationship between several variables. VAR** is most suited for when each of the variables is to be predicted before predicting the target.

The time series we are looking into is multivariate timeseries but also additionally has multiple regions and is most suited as a **supervised** learning, regression, prediction task. The variables are **independent** variables and we can break the data into train and test sets.

DATASET

Groundwater, which is found in aquifers below the surface of the earth, is one of our most important natural resources. Groundwater provides drinking water for a large portion of California, nay, the nation's population. It also supplies business and industries, and is used extensively for irrigation. California depends on groundwater for a major portion of its annual water supply, particularly during times of drought. This reliance on groundwater has resulted in overdraft and unsustainable groundwater usage in many of California’s basins, particularly so in the San Joaquin River basin.

![What is groundwater](../images/groundwater.png)

**What is groundwater?**

Groundwater is water that exists underground in saturated zones beneath the land surface. The upper surface of the saturated zone is called the water table. Groundwater is a part of the natural water cycle. Some part of the precipitation that lands on the ground surface infiltrates into the subsurface. The part that continues downward through the soil until it reaches rock material that is saturated is groundwater recharge. Water in the saturated groundwater system moves slowly and may eventually discharge into streams, lakes, and oceans. An aquifer is a body of rock and/or sediment that holds this groundwater.

The water level in an aquifer that supplies water to a well does not always remain the same. Factors affecting groundwater levels that are studied in this project include:

1. Droughts  
2. Seasonal variations in precipitation  
3. Reservoir levels  
4. Pumping for human needs such as domestic, agriculture and industrial

If water is pumped at a faster rate than an aquifer is recharged by precipitation or other sources  
of recharge, water levels drop. This can happen during drought, due to the extreme deficit of rain.

The analysis is performed against the backdrop of the [Sustainable Groundwater Management Act](https://water.ca.gov/programs/groundwater-management/sgma-groundwater-management) that was passed in 2014 in California. SGMA requires locals agencies to form groundwater sustainability agencies (GSAs)for the high and medium priority basins.  
  
Each dataset is acquired at the available granularity and transformed to features at a township-range and year granularity.

Graphical user interface, text

Description automatically generated

ACQUIRING DATA AND CLEANING

Provide links to each dataset ETL documentation.

FEATURE PREPROCESSING USING SKLEARN PIPELINE

Graphical user interface, text, application, Word, email

Description automatically generated

[**In the notebook,**](https://deepnote.com/workspace/simi-talkar-1cb943f3-53ab-4def-8a04-90ed01b36309/project/Milestone-2-Water-Wells-b042e2da-6536-449d-95b8-d85fa08825de/%2Fmilestone2_waterwells_deepnote%2Fml%2Fbasic_transforms_set_baseline-StandardScaler.ipynb) **click on each box above to see further details.**

Chart, histogram

Description automatically generated

*We have several variables that are null as seen in the chart above. A pipeline was created using SKLearn to perform the following imputations and transformations.*

**Vegetation dataset  
Cause**We do not have data for the above vegetation after 2014 for any of the TownshipRanges. That is, it is absent for the years [2015, 2016, 2017, 2018, 2019, 2020, 2021]. **Imputation**  
We will be making an assumption that the vegetation landscape of a TownshipRange does not radically differ from year to year. The vegetation values will carry over from 2014 for these vegetation types for each of the township ranges. The imputation is carried out by finding the mean of the data for all Townships. This will be the only available data in the year 2016. This is then filled into the rest of the years for each TownshipRange using a merge.

**Crops dataset  
Cause**We only data for 3 years over the analysis survey, 2014, 2016, 2018 **Imputation**   
We assume little year-to-year variation in crop farming and extended the data for the missing years (2015, 2017, 2019) with the data from the previous years (i.e. we assume that 2015 crops = 2014 crops). We understand that farming practices like crop rotation would challenge these assumptions. The imputation is carried out by sorting the dataframe by year and then township and then forward filling the data over to the years following. **Soils dataset  
Cause**  
We have data for soils for 2016 and not for other years. **Imputation**We will be making an assumption that the basic nature of the soil of a TownshipRange does not radically differ from year to year. As we do not expect the soil type to change from year, the 2016 soil data are used for all the other years. The imputation is carried out by finding the mean of the data for all Townships. This will be the only available data in the year 2016. This is then filled into the rest of the years for each TownshipRange using a merge.

**Well Completion dataset  
Cause**  
Absence of well completion reports filed in township range in certain, or all years causes values to be null.  
**Imputation**We impute the GROUNDSURFACEELEVATION\_AVG for a TownshipRange as the mean of the values over all years for that TownshipRange. This value that is stored in the data is the average of the elevation of the ground where wells were constructed in that TownshipRange and hence elevation was known.

But this does not cover all the NaNs since 76 TRs do not have any value at all and they should be filled with the average elevation of the entire area.

For the other variables since they are the average of the well yield, static water level, top of perforated interval, bottom of perforated interval and total completed depth of the wells constructed in each TownshipRange and so if there are no wells newly constructed, in a township range in a certain year, this value will be set to 0 as these variables are well specific measurements.

**Reservoir dataset   
Cause**Reservoir data is available from the California reporting stations from 2018, we are missing the data from 2014 through 2017.  
**Imputation**  
Since in these years, California was stricken with drought, we will impute the values to be the minimum of the PCT\_OF\_CAPACITY for that TownshipRange.

**Population dataset  
Cause**Population density is missing for the final year of 2021 for all townships. **Imputation**  
We will be using the previous years(2020's) trend over the year 2019 and add to the previous years (2020) value for each TownshipRanges for the missing values in year 2021.

**Challenges faced :** Pipeline steps are executed serially, where the output from the first step is passed to the second step, and so on. ColumnTransformers are different in that each step is executed separately, and the transformed features are concatenated at the end. By default, any columns you pass into the ColumnTransformer that aren’t specified to be transformed will be dropped (remainder='drop'). If you have columns that you want to include but do not need to be transformed, specify remainder='passthrough'.

CHOICE OF EVALUATION SCORE METRIC

Baseline

Since this is a regression problem with continuous variables, the evaluation scores find the difference (error) between the predicted value and the observed value. Some of the choices are **M**ean **S**quare **E**rror, **R**oot of the **M**ean **S**quare **E**rror and **R**oot **M**ean **S**quare **E**rror. We can also look into the mean magnitude of the errors in a set of predictions (**M**ean **A**bsolute **E**rror). These are negatively oriented scores which means lower scores are better. Also, the scores where the errors are squared give a higher weight to large errors and used when large errors are undesired. They ignore the direction of error.

R2 (R squared), MSE (Mean Squared Error) or RMSE (Root Mean Squared Error)

For the initial comparison of algorithms, before hyper tuning and for generating a list of models for PyCaret to compare, R squared gave us a quick indication of the relative performance and was relied on. R2 provides the proportion of the variance for the target that’s explained by selected features in the model. It is independent of the scale of the features and ranges from 0 to 1. A negative value implies worse than mean model. When evaluating an algorithm, it is prudent to look into multiple regression scores and not just R-squared since the acceptable threshold of the error will depend on the distribution of the target value itself. To keep the groundwater depth prediction difference from the mean in check we additionally looked into MSE and RMSE.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **R2 Score** | **MSE** | **RMSE** |
| Dummy Regressor | -0.0571 | 17442.601 | 132.070 |
| Extra Tree | 0.4340 | 6499.055 | 68.117 |
| XGBoost |  |  |  |
| SVM |  |  |  |

SCALING

Many elements used in the objective function of a learning algorithm (such as the RBF kernel of Support Vector Machines or the L1 and L2 regularization of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

**IMPACT OF SCALING WITH MINMAX OR STANDARD SCALER**

1.There is a difference in clustering when the features are scaled using MinMaxScaler (no negative values for the range set) versus when they are scaled using StandardScaler (required for algorithms such as SVM) . Features scaled using MinMaxScaling do not show the large area of incorrectly clustered data points.

Chart

Description automatically generated

Figure : Clustering with features scaled using StandardScaler

Chart, scatter chart

Description automatically generated

Figure : Clustering with features scaled using MinMaxScaler

We are considering a region within a state that is a known agricultural area. It is very likely that the features in the area that contribute to the land being fertile and suitable for agriculture (among others) is resulting in one major cluster, with the other not as significant.  
2. Major difference in number of components that explain 70% of the variation when difference is observed scaling of the feature is performed to be within the [MinMaxScaler](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html) ( range of [0, 1] ) as compared to features scaled with [StandardScaler](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html) (remove the mean and have unit variance). 3.The variations in the evaluation scores and PCA number of components explaining 70% of the variation is pronouncedly different and so the two scaling methods were evaluated in different notebooks.   
4. With Standard Scaling, the Biplot is difficult to read, but the components make clear sense.  
5. The Silhouette score in clustering is better with StandardScaler.  
6. The evaluation metrics (R2, MSE, RMSE) for testing improve with StandardScaler.  
7.With MinMaxScaling, the Silhouette Score does not show as much negative area for the first three clusters.

**How much does each of the features contribute to the variance and to the prediction?**

CORRELATION BETWEEN FEATURES

Graphical user interface, chart, scatter chart

Description automatically generated

We are considering features with absolute value of 0.7 as being highly correlated.  
Well construction specific correlations that can be considered as expected since each value describes the well's dimensions and capacity.

To the right of the heatmap, we see TOPOFPERFORATEDINTERVAL and BOTTOMOFPERFORATEDINTERVAL highly correlated.

BOTTOMOFPERFORATEDINTERVAL and TOPOFPERFORATEDINTERVAL are heavily correlated with TOTALCOMPLETEDDEPTH, while TOTALCOMPLETEDDEPTH is correlated with STATICWATERLEVEL

**Well Count**

All well counts, further categorized as Domestic, Agriculture and Public form highly correlated categories (WELL\_COUNT --> (WELL\_COUNT\_DOMESTIC, WELL\_COUNT\_AGRICULTURE))

**Crops**

We observe some few other correlations between some type of crops (e.g. 0.57 between CROP\_D16 plums, prunes or apricots and CROP\_D5 which is peaches/nectarines and between some crops and soils (e.g. 0.61 between CROP\_T31 potatoes and sweet potatoes and SOIL\_ENTISOLS\_A soil of hydrographic group A corresponding to coarse texture soils with high saturated hydraulic conductivity. This correlation also makes sense since potatoes tend to not grow well in soils retaining too much water.

**Vegetation**

We also see some level of correlation between the different type of forest vegetation. For example VEGETATION\_CANYON\_LIVE\_OAK, VEGETATION\_HARD and VEGETATION\_KNOBCONE\_PINE show correlation values between 0.46 and 0.6. This might be explained by the fact that some type of vegetation tends to found frequently together.

VARIANCE USING PCA BIPLOT

After observing the feature correlation in the heatmap, we decided to check for latent features in the dataset that will effectively combine the correlated features, and create a new set of features that are a weighted linear combination of original features, using PCA. The number of components selected explain **70%** (a threshold we selected based on the Scree plot), of the original variance of the dataset.

The first component contains all features describing a well in an area.  
The second component includes a linear combination of well counts and population in an area.  
The third component is largely about precipitation and reservoir capacity, along with ground surface elevation.

Top 4 most important features in each component

===============================================

Component 0: ['TOTALCOMPLETEDDEPTH\_AVG', 'BOTTOMOFPERFORATEDINTERVAL\_AVG', 'TOPOFPERFORATEDINTERVAL\_AVG', 'STATICWATERLEVEL\_AVG']

Component 1: ['WELL\_COUNT\_DOMESTIC', 'WELL\_COUNT\_PUBLIC', 'WELL\_COUNT\_AGRICULTURE', 'POPULATION\_DENSITY']

Component 2: ['PCT\_OF\_CAPACITY', 'AVERAGE\_YEARLY\_PRECIPITATION', 'GROUNDSURFACEELEVATION\_AVG', 'WELL\_COUNT\_AGRICULTURE']

Component 3: ['GROUNDSURFACEELEVATION\_AVG', 'WELL\_COUNT\_INDUSTRIAL', 'AVERAGE\_YEARLY\_PRECIPITATION', 'POPULATION\_DENSITY']

Component 4: ['WELL\_COUNT\_INDUSTRIAL', 'AVERAGE\_YEARLY\_PRECIPITATION', 'POPULATION\_DENSITY', 'WELL\_COUNT\_AGRICULTURE']

Component 5: ['POPULATION\_DENSITY', 'GROUNDSURFACEELEVATION\_AVG', 'TOTALDRILLDEPTH\_AVG', 'WELL\_COUNT\_AGRICULTURE']

Component 6: ['TOTALDRILLDEPTH\_AVG', 'WELLYIELD\_AVG', 'STATICWATERLEVEL\_AVG', 'GROUNDSURFACEELEVATION\_AVG']

Projections IN THE NEW PCA SUBSPACE

Graphical user interface, chart, scatter chart

Description automatically generated

Figure 3 PCA Biplot

The PCA biplot emphasizes the collinearity we saw in the heatmap (see in notebook) in features related to a well such as completed depth, top and bottom of perforations (for filter).

Crop D12 (almonds) and Alfisols\_D soil are correlated (small angle between them), where as Crop\_D12 “pistachios” are quite different as a feature with a greater than 90 degrees angle between pistachios and almonds. As per [USDA soil taxonomy](https://dbpedia.org/page/Alfisol), because of their productivity and abundance, the Alfisols represent one of the more important soil orders for food and fiber production. They are widely used both in agriculture and forestry,

FEATURE CORRELATION IN TRAIN AND TEST SET

It can be seen that the feature importance predicted in model aligns with the feature to target correlation in the train set. But the correlation in the test set predicts a different order of feature importance.

FEATURE IMPORTANCE IN PREDICTION

**SHAPELY VALUES (Shapely Additive Explanations) : Model Agnostic**

Graphical user interface, application

Description automatically generated

Figure : Shapley Values

Shapely values are a commonly used machine learning technique to explain a model. It is **model agnostic.** In Pycaret, we use the **interpret\_model(best)** thatprovides the above plot.  
[Shapely works by changing one feature at a time](https://www.youtube.com/watch?v=NBg7YirBTN8&t=301s), taking into account the probability of creating improbable samples and isolated effects. The target is predicted and compared as a difference to the mean of all original predictions. The expected value of these differences is the Shapely value of this feature.

Ground Surface Elevation of the area is the more important indicator followed by depth of wells and current depth of water in the wells in the area and some soils such as the “alfisols” which is the fertile soil that supports San Joaquin valley’s reason for being an agricultural belt, Entisols and Mollisols, which are soils of grassland ecosystems. Among crops, Pistachios (“D14”) and alfalfa and alfalfa mixtures ("P1") carrying areas, influence predictions.

[The SHAP plot](https://www.youtube.com/watch?v=ZkIxZ5xlMuI) also shows the positive and negative relationships of the predictors with the target variable with 0.0 as the midline in the graph. The less (more blue) the groundsurface elevation and afisols, the more the depth. The more (more red) the depth of the wells, the more the ground water depth (kind of intuitive). More pistachios, more depth, but more alfalfa, predicts, less depth.

**FEATURE IMPORTANCE FROM THE BEST MODEL : Model Specific**

Chart, scatter chart

Description automatically generated

Figure : Feature Importance for ET regressor

Feature importance in the random tree boosting algorithm Extra Trees also afforda Groundsurface Elevation the most importance along with well specific feature of the depth of the well and current level of water in the wells in the area. In addition to this, Alfisol\_D (the fertile soil) and Pistachios also feature prominently in the prediction. Reservoir capacity is considered an important feature by this algorithm.

## **EVALUATING VARIOUS ALGORITHMS AND CHALLENGES FACED IN PREDICTIONS**

[Supervised Learning Manual (without PyCaret)](https://deepnote.com/workspace/simi-talkar-1cb943f3-53ab-4def-8a04-90ed01b36309/project/Milestone-2-Water-Wells-b042e2da-6536-449d-95b8-d85fa08825de/%2Fmilestone2_waterwells_deepnote%2Fml%2Fsupervised_learning_manual.ipynb)

We followed the following strategies to improve the evaluation score.

**Normalizing the Target**

|  |  |
| --- | --- |
| **Chart, histogram  Description automatically generated**  Figure : Raw Target | **Chart, histogram  Description automatically generated**  Figure :Normalized Target |

**SVM** was the first linear algorithm to provide non-negative test evaluation scores, but to improve the training scores, in SVM, we employed SKLearn’s **TransformedTargetRegressor**, with numpy’s square root function, that can be used to wrap any SKLearn ML algorithm class and has the following advantages:

a. You do not have to call **inverse\_transform** every time after prediction.  
b. You can specify the transformation and inverse transformation function in the initialization.  
c. The wrapped regressor can be used in cross validation and parameters can be set using the wrapping classes’s parameters deduced by calling model.get\_params().keys().   
d. The wrapper does not have feature importance property though a wrapper class was created (for TransformedTargetRegressor) that extracts the feature\_importance\_ from the parent.

2. High training score and very low test score was obtaining after tuning XGBoost an industrial strength algorithm that returned extremely high training scores .  
3. Increasing number of folds in Pycaret upto exactly 51, improved R-squared, MSE and RMSE.

Other strategies we tried out with no effect or led to reduction in scores:

1. Added cluster label as a feature after applying unsupervised learning to create clusters with highest Silhouette Score. Unsupervised learning indicated that the best clustering (in terms of inter and intra cluster distance) is obtained with two clusters. We created a feature by labeling each sample data with one of the two clusters.

2. Reduced features using PCA by creating a new dataset in the PCA feature space.

As seen in the table below, little to no improvement was seen in adding cluster label as a feature or by using PCA transformed data. Some decrease in metrics was observed as well.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **R2 Score (Train)** | **MSE (Train)** | **RMSE (Train)** | **R2 Score(Test)** | **MSE (Test)** | **RMSE (Test)** |
| Unnormalized SVM (no clustering) | 0.462 | 8879.232 | 94.230 | 0.150 | 14250.850 | 119.377 |
| Normalized SVM (no clustering) | **0.526** | **7817.253** | 88.415 | 0.163 | 14031.817 | 118.456 |
| Unnormalized SVM **(with clustering)** | 0.465 | 8821.246 | 93.921 | 0.160 | 14082.799 | 118.671 |
| Normalized SVM (with **clustering)** | 0.524 | 7857.708 | 88.643 | 0.168 | 13942.948 | 118.080 |
| Unnormalized SVM (on PCA transformed data) | 0.382 | 10200.413 | 100.997 | 0.134 | 14515.185 | 120.478 |
| Normalized SVM (on PCA transformed data) | 0.460 | 8905.953 | 94.371 | 0.134 | 14518.08 | 120.490 |

**TREE BASED ALGORITHM XGBOOST**

Since the RandomForestRegressor (tree based algorithm) indicated a high training score, we tried XGBoost with sckit-learn’s wrapper class XBGRegressor. This was tried mainly on the normalized target. We also used hypertuning to improve regularization. We tried out different learning rates since in boosting algorithms, the prior stump’s evaluation leads to a better next step, we also tried a range of lambda (regularization) and gamma (for tree pruning threshold) values with cross validator RandommizedSearchCV. The evaluation metric was set as RMSE and an evaluation set of both train and test set were provided.

The TransformedTargetRegressor was further wrapped in a class which calls the parent (XBGRegressor’s) feature importance property.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **R2 Score (Train)** | **MSE (Train)** | **RMSE (Train)** | **R2 Score(Test)** | **MSE (Test)** | **RMSE (Test)** |
| Normalized XBBoost | 0.688 | 5140.892 | 71.700 | -0.227 | 20571.901 | 143.429 |

Feature Importance with XGBoost indicates that GROUNDSURFACEELEVATION\_AVG as the top predictor, followed by pasture crops, ALFISOLS which is the fertile soil in the valley and well features like completed depth and water level in well.

Calendar

Description automatically generated with medium confidence

While XGBoost dramatically improved the train set scores, it shows poor generalizability to unseen data as the test scores are very poor.

**KNN ALGORITHM**

Hypertuning a K Nearest neighbors algorithm on botg unnormalized target and normalized target gave us the optimal k as 20. This algorithm had better generalization than XGBoost but did not have train or test scores as good as SVM. Moreover, in KNN, normalizing the target was slightly detrimental to the evaluation metrics for the train set. Using the PCA reduced dimensional dataset reduced the evaluation scores for the training set but improved those for the test set.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **R2 Score (Train)** | **MSE (Train)** | **RMSE (Train)** | **R2 Score(Test)** | **MSE (Test)** | **RMSE (Test)** |
| Unnormalized KNN | 0.654 | 5713.425 | 75.587 | 0.029 | 16287.523 | 127.623 |
| Normalized KNN | 0.641 | 5915.180 | 76.910 | 0.094 | 15191.470 | 123.253 |
| Unnormalized KNN (with PCA) | 0.589 | 6772.706 | 82.296 | 0.040 | 16094.832 | 126.865 |
| Normalized KNN (with PCA) | 0.577 | 6970.285 | 83.488 | 0.087 | 15311.969 | 123.742 |

EXPLANATION OF THE ALGORITHMS TO HELP HYPERTUNE

SUPPORT VECTOR MACHINE REGRESSION

Kernelized Support Vector Machines provide complex models that go beyond linear decision boundaries. They take the original input dataspace and transform it to a new higher dimensional feature space where it becomes much easier to classify or regress the transformed data using a linear hyperplane classifier. Support Vector Machine finds a hyperplane in the new feature space to create linear decision boundaries to separate the data. The distance from the hyperplane to the closest data point is called the margin. The data points from each class closest to the margin classifier are called support vectors. For the radial basis function kernel, the similarity between two points in the transformed feature space, is an exponentially decaying function of the distance between the vectors in the original input space. The RBF kernel has a parameter gamma that controls how far the influence of a single training example reaches, which in turn affects how tightly the decision boundaries end up surrounding points in the input space. Small gamma means more generalization since, points farther apart are considered similar which results in more points being grouped together and smoother decision boundaries. SVMs also have a regularization parameter, C, that controls the trade-off between satisfying the maximum margin criterion to find a simple decision boundary. If gamma is large, then C will have little to no effect. While if gamma is small, the model is much more constrained and the effect of C will be similar to how it would affect a linear classifier. Typically, gamma and C are tuned together with the optimal combination, typically in an intermediate range of values.

XGBOOST

It is E**X**treme **G**radient **Boost**ing that is best suited for large complicated datasets.

Understanding Boosting

A family of models with production level implications. When we learn a final predition function F(x), it will be a sum of  **underpowered sequential weak learners**  (these weak learners could be any linear  
regression/svm/tree).

STEP 0

    1. Define your loss function. True label and y-hat (prediction). This function should be differentiable.

STEP 1

    1. Start with extremely weak learner (F1(x)). Initially, say it is the mean of the y targets.  
    2. This is the initial prediction.  
    3. The residuals (sum of difference of each data point from the mean, predicted by the current model  
 line, define how  good or bad this initial prediction is.

STEP 2    This is essentially gradient descent.

    At every iteration we add the previous weak learner to a eta\*(new learner).  
 That is, we move in the direction of the decreasing loss by a factor of a learning rate (eta).  
 Every week learner descends down the gradient by reducing the loss.

* Start with all residuals in a leaf node
* Calculate the Similarity Score for this leaf which is the Square of the Sum of the Residuals
* divided by (number of Residuals + lambda). Lambda is the regularization parameter that minimizes dependence on any one feature.
* Note the residuals are not sqaured before summing.Some of the residuals might cancel each other out.
* Now the question is : Can we do a better job if we cluster resdiuals together in some manner.(create a line separating the clusters)
* We split the observations into two parts based on say an average of a feature.
* Calculate the similarity score of each leaf node with residuals clustered into the two nodes.
* The Gain is the sum of the similarity scores of these two leaf nodes minus the similarity score of the "root" node.
* We keep the clusters that result in the largest Gain.
* Pruning trees. Set a threshold: Tree Complexity Parameter (gamma) to compare Gains against. If the Gain is greater than gamma, then the tree is not pruned. We start with the lowest tree.
* Lambda, the regularization parameter when set to more than 0, also can cause tree pruning, since it reduces the Similarity Scores and hence the Gain (which is compared to the threshold Gamma)
* Lambda has a greater effect on tree pruning than Gamma.
* After the trees have been created, we calculate the Outcome.Outcome is the Sum of Residuals/(number of residuals + Lambda)
* Once we have the Outcomes we can make new predictions. Learning Rate = ETA

= original prediction (typically mean) + Learning Rate x Output

[**EXTREME(LY) RANDOMIZED TREE ALGORITHM**](https://www.youtube.com/watch?v=ULFDltLnR6U)

**Basic Principle of Random Forests that Extremely Randomized Tree algorithm is similar to:** Bootstrapping + Aggregating (Bagging) to create random trees characterized by repeated sampling with replacement from the dataset and randomly using a subset of columns in each tree and finally aggregating the results of all trees. Evaluation is carried out on the out-of-bag samples (those that did not go through a tree) and evaluating prediction errors.

Difference in Extra Tree from Random Forest:

1. Splits in Random Forest are evaluated based on purity pf samples and GINI Index, where as in ET, splits are entirely random. Leads to better speed.  
2. Whole samples considered for each tree (no bootstrapping).  
3. Faster runtime encourages use with datasets with large number of attributes and noisy features.  
4. Randomly splitting by attributes can lead to smaller sample size available for a tree and hence bias.  
5. Randomness also generally leads to less interpretability.   
6. Less computational power than random forest.

[**K-NEAREST NEIGHBORS ALGORITHM**](https://www.youtube.com/watch?v=ULFDltLnR6U)

KNN requires us to specify the number of neighbors in the proximity of a data values that must be considered when classifying or forming a regression model.

CHALLENGES WITH THE LOW EVALUATION METRIC SCORE

While we were able to score a very high r-Square for the training set, even with hypertuning, we were not able to obtain a reasonable score for the test set. Below the top correlation coefficients of target GSE\_GWE can be seen in relation to the features. The Train set is on the left and the test set is on the right. It can be seen that the feature importance seen in the Shapely and for the Extra Tree regressor, such as GROUNDSURFACELEVEVATION\_AVG, well features such as STATICWATERLEVEL and depth of well as well as VEGETATION\_CALIFORNIA\_COAST\_LIVE\_OAK predominantly play a role in the relation to the target in the train set. While in the test set, it is precipitation that has the most correlation followed by all the well characteristics. It is possible that this difference is contributing to the low evaluation score.

|  |  |
| --- | --- |
| Table  Description automatically generated  Figure :Train set correlation with target GSE\_GWE | Table  Description automatically generated  Figure :Test set correlation with target GSE\_GWE |

HYPERTUNING

GridSearchCV and RandomizedSearchCV were employed by creating a parameter grid of parameters that can be tuned in each instance of a chosen algorithm. In PyCaret automated process as well hyper tuning was used but tuning did not always result in a better R-squared or RMSE score. Tuning in the case of K- Nearest neighbors algorithm

[**PyCaret: SuperVISED LEARNING AUTOMATED**](https://deepnote.com/workspace/simi-talkar-1cb943f3-53ab-4def-8a04-90ed01b36309/project/Milestone-2-Water-Wells-b042e2da-6536-449d-95b8-d85fa08825de/%2Fmilestone2_waterwells_deepnote%2Fml%2Fsupervised_learning_automated.ipynb)

After going through various machine learning algorithms manually, one at a time and hypertuning each, we found the usage of PyCaret extremely beneficial. In addition to having the ability to consider [multiple algorithms](https://pycaret.readthedocs.io/en/latest/api/regression.html?highlight=create_model#pycaret.regression.create_model), PyCaret allows for single line code to  
a) hypertune parameters with cross validation  
b) to conduct PCA and target normalization [with parameters in the setup](https://pycaret.readthedocs.io/en/latest/api/regression.html?highlight=create_model)  
c) it allows for feature\_selection (by ignoring features) and feature\_engineering (by specifying interactions, polynomial features), also with setup parameters.

After setting up the experiment that can/will be logged in the open source MLFlow tool (see how to view the experiments in section below), compare\_models will compare all, or a specified list of models with cross validation folds, optionally specified. The evaluation metrics are duly returned based on whether the task is classification or regression and the tabulated data is sorted with the best model at the top.

Pycaret brought to fore extremely random tree (ET) algorithm when models were compared and with folds = 51, the r-squared and MSE improved further.

With Feature Engineering, was tied out as well. New features are created based on all polynomial combinations that exist within the numeric features in a dataset to the degree defined in polynomial\_degree param. Sometimes the relationship between dependent and independent variables is more complex. Creating new polynomial features sometimes might help in capturing that relationship which otherwise may go unnoticed.

PyCaret can also create a complete pipeline of transforming the features including imputing nulls but since we used scikit-learn pipeline for this purpose, we set this parameter to False.

SETUP PARAMETERS IN PYCARET FOR REGRESSION

Transform target is performed by PyCaret using transform\_target parameter. The default method for this transformation is based upon the task of algorithm. ‘Box-Cox’ (strictly positive data) and ‘Yeo-Johnson’ (positive or negative data) are the possible methods of transformation.

This has the single largest effect in improvement pf evaluation metrics. This was seen in SVM and Pycaret in the most stark fashion.

Best results were obtained where these feature selection values were set

transform\_target = true (default box-cox)  
 remove\_multicollinearity = True,  
 multicollinearity\_threshold=0.1  
 #Low Variance in feature  
 ignore\_low\_variance=True,

fold was set to 51

PyCaret was also applied on PCA transformed data but it did not help with the evaluation metrics. After running compare\_models through all available models in PyCaret, we drew down from the top list to Extra Trees, Light Gradient Boosting Machine, Random Forest, XGBoost and K Nearest Neighbors algorithm. The scores tabulated are seen below with Extra Tress providing the best mean score for 51 folds of cross validation.

Table

Description automatically generated

We found that the best model does not perform very well on the unseen data but overfits the training set.

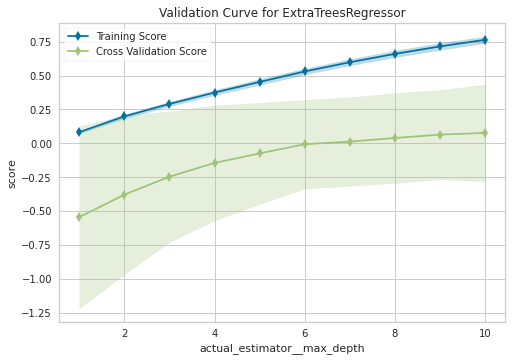
Tuning the best ET model also reduces the performance. The r-squared drops for 0.43 to 0.02 and the MSE increases from 49.84 to 72.37

Table

Description automatically generated

Chart, scatter chart

Description automatically generated



Chart, scatter chart

Description automatically generated

APPENDIX: ML FLOW

Mlflow let’s one view the experiments in a UI.In the notebook run

!mlflow ui –host=localhost –port=5000

While that cell is running, in the next cell run

mlflow ui

Then in a browser window, type “localhost:5000”

This opens up a window as seen below that helps us compare experiments.

[Note: I could run this in my local environment, but not in Deepnote]

Graphical user interface

Description automatically generated