**Module 3 - Fundamentals of AI**

**Course Number:** EAI 6000

**Academic Term:** Fall 2020 CPS Analytics  
**Instructor’s Name:** Kasun Samarasinghe

**Assignment Completion Date:** 11-14-2020

**Submitted By:**

Dhiren Vasudev Pagrani

Sunil Raj Thota

Shivani Sharma



**INTRODUCTION**

**Decision Trees** are created by an algorithmic method that classifies ways to split a data set based on various conditions. It is one of the most extensively used and real-world approaches for supervised learning.

These are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to produce a model that envisages the value of a target variable by learning simple decision rules inferred from the data features. The rules are generally in form of if-then-else statements. Deeper the tree, the more complex the rules and fitter the model.

**Random Forest** is a method used in modeling predictions and behavior analysis and is built on decision trees. It contains many decision trees that signify a discrete instance of the classification of data input into the random forest. This technique takes consideration of the instances individually, taking the one with the majority of votes as the selected prediction. Each tree in the classifications takes input from samples in the initial dataset. Features are then randomly selected. In such a way, it enables any classifiers with weak correlations to create a strong classifier.

The random forest associates hundreds or thousands of decision trees, trains each one on a somewhat diverse set of the observations, segregating nodes in each tree considering a limited number of the features. The final outcomes of the random forest are made by averaging the predictions of each distinct tree.

**Gradient Boosting Method** is used to predict the action of drugs by plotting decision trees of previous results and GBR would predict second trees and so on. The name gradient boosting arises because target outcomes for each case are set based on the gradient of the error with respect to the prediction.  In boosting, each new tree is a fit on a modified version of the original data set. The gradient boosting algorithm can be easily explained by first introducing the AdaBoost Algorithm. The AdaBoost Algorithm begins by training a decision tree in which each observation is assigned an equal weight. After evaluating the first tree, we increase the weights of those observations that are difficult to classify and lower the weights for those that are easy to classify. The second tree is therefore grown on this weighted data.

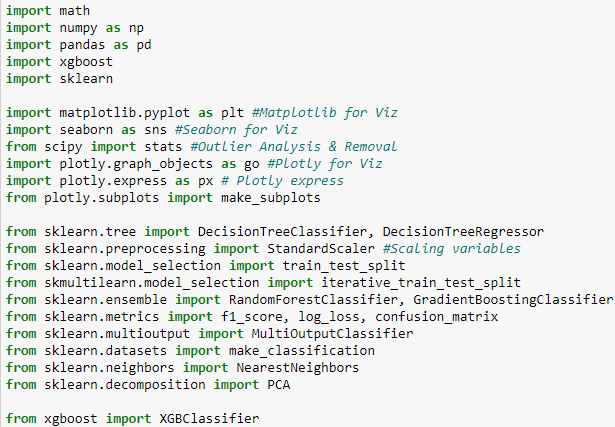
The idea here is to improve the predictions we got from the first tree. Therefore, our new modelTree 1 + Tree 2. We then compute the classification error from this new 2-trees ensemble model and grow a third tree to predict the revised residuals. We repeat this process for a specified number of iterations. Subsequent trees will help us to classify observations that were not well classified by the previous trees. Like this, predictions from the final ensemble model is represents the weighted sum of the predictions made by the previous tree models.

**XGBoost** also known as Extreme Gradient Boosting is a scalable end-to-end tree boosting system which is mainly used for implementation of gradient boosted decision trees designed for speed and performance. XGBoost has high predictive power and is almost 10 times faster than the other gradient boosting techniques. It also includes a variety of regularization techniques which reduces overfitting and improves overall performance. Some of the benefits of XG Boost are as follows:

* Regularization
* Automatically handles missing values
* Built-in Cross Validation
* Tree Pruning

**ANALYSIS**

Here is the analysis for the Decision Tree, Random Forests, GBM and XG Boost algorithms. Following are the libraries we need for the above algorithm’s analysis on our selected dataset:

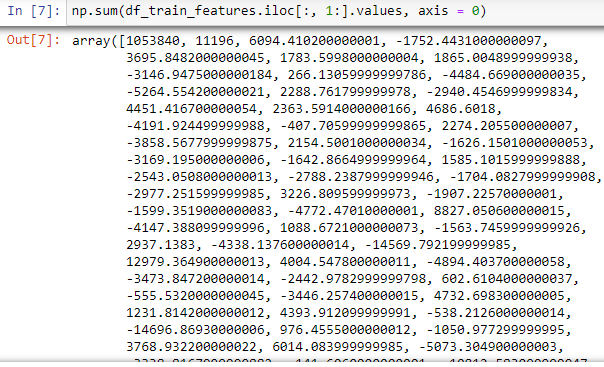


Now following is the logic for storing the dataset in data frames:

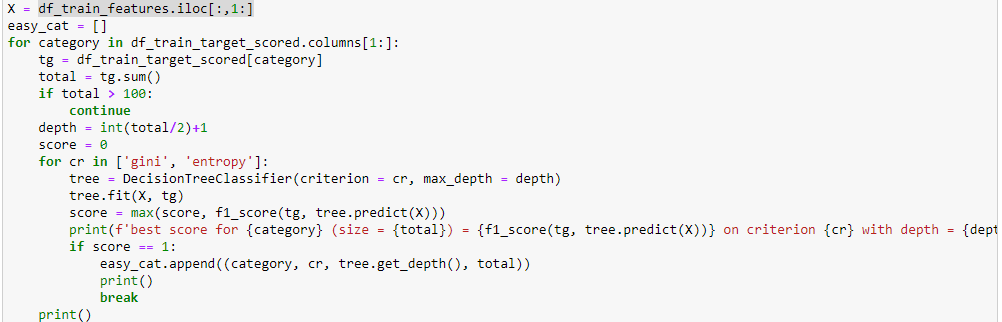


The 2 most popular decision tree’s decisions are Gini Index and Information Entropy. The gini score is always the same because they always add to 0 in the formula above. A gini score of 0 is the purest score possible.

Entropy is more computationally heavy due to the log in the equation. Like Gini, the basic idea is to measure the condition of a grouping by the target variable. Instead of utilizing simple probabilities, this method takes the log base2 of the probabilities. The entropy equation uses logarithms because of many advantageous properties. The Main advantage is the additive property it provides.

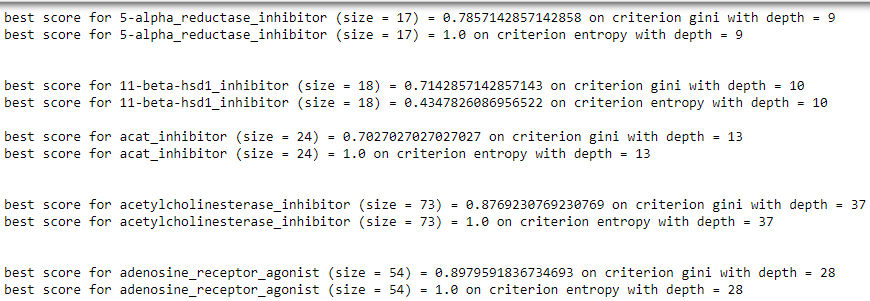


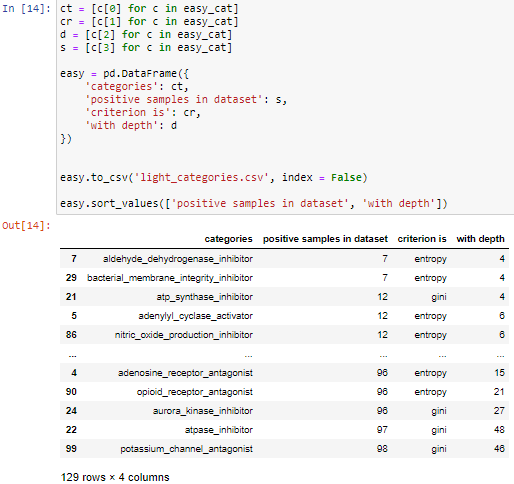
Here is the array of df\_train\_features sum values



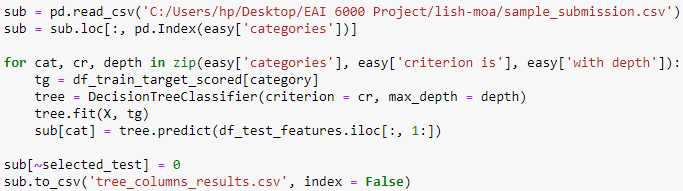
**Gini = 1 – pj2**

**Entropy = - pj log2 pj**





Categories, positive samples in dataset, criterion is, and with depth are called as ct, s, cr, and d.



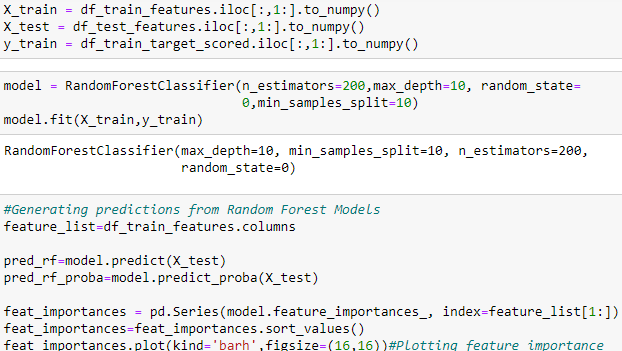
From this we have obtained the predicted values and stored as tree\_columns\_results.csv file

**Advantages of Decision Trees:**

* Good for interpreting data in an exceedingly highly visual way
* Can handle a mixture of numerical and non-numerical data
* Easy to define rules, e.g. ‘yes, no, if, then, else…’ and can handle multidimensional data
* Can be easily combined with other decision-making techniques

**Disadvantages of Decision Trees :**

* Overfitting can become a problem if a decision tree’s design is too complex
* It is easy for outcomes to be biased in favor of the dominant class
* They aren't well-suited to continuous variables
* Provide lower prediction accuracy compared to other predictive algorithms



Random Forest is a tree-based machine learning algorithm that leverages the facility of multiple decision trees for creating decisions. It's a forest of randomly created decision trees. Each node within the decision tree works on a random subset of features to calculate the output. The random forest then combines the output of individual decision trees to get the ultimate output. This process of mixing the output of multiple individual models is called as Ensemble Learning.



Here’s the plot between list of features and following things are observable from the plot above - The first principal components for c- and g- variables have the highest feature importance. As observed in our EDA - dosage, treatment type and duration of treatment have the lowest feature importance in the dataset. c- and g- PCs solely contribute to the performance of the model.

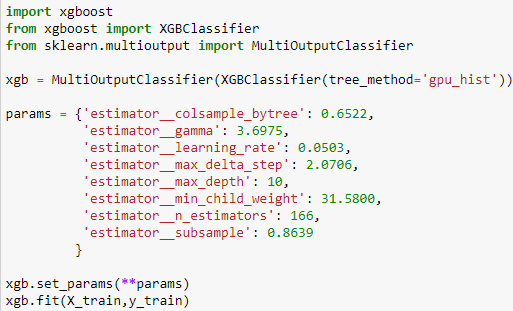
**Advantages of Random Forest:**

* Missing values are substituted by the variable appearing the foremost in a particular node
* Good performance - little tuning required
* Built-in validation set
* Provides the best accuracy and can automatically balance data sets
* No pre-processing required robust to outliers

**Disadvantages of Random Forest:**

* Can become slow on large data sets and Less Interpretable
* Although accurate, often cannot compete with advanced boosting algorithms

In XGBoost, we fit a model on the gradient of loss generated from the previous step. In XGBoost, we just modified our gradient boosting algorithm so that it works with any differentiable loss function.



**Advantages of XG Boosting:**

* Fast to interpret and Outliers have minimal impact.
* Handles large sized datasets well and Less prone to overfitting
* Good Execution speed and Good model performance

**Disadvantages of XG Boosting:**

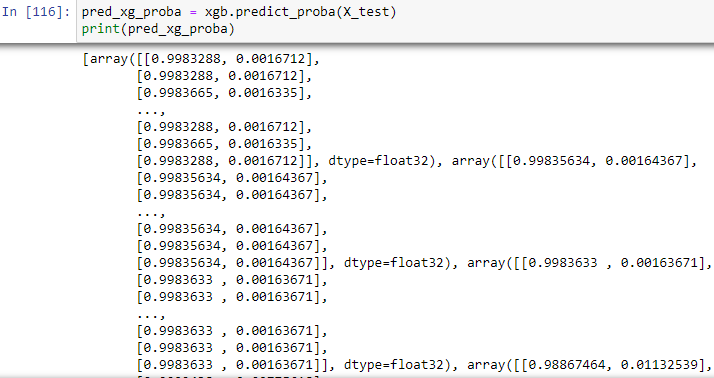
* Difficult interpretation
* Overfitting possible if parameters not tuned properly.
* Visualization is tough
* Harder to tune as there are too many hyperparameters.



Bagging and boosting are two widely used ensemble learners. Though these two techniques can be used with several statistical models, the most predominant usage has been with decision trees.

Here we can see the predicted values by the XG Boosting model.

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.



We first model data with simple models and analyze data for errors. These errors signify data points that are difficult to fit by a simple model. Then for later models, we particularly focus on those hard to fit data to get them right. In the end, we combine all the predictors by giving some weights to each predictor.

**Advantages of Gradient Boosting:**

* High-performing
* Often provides predictive accuracy
* Lots of flexibility
* No data pre-processing required

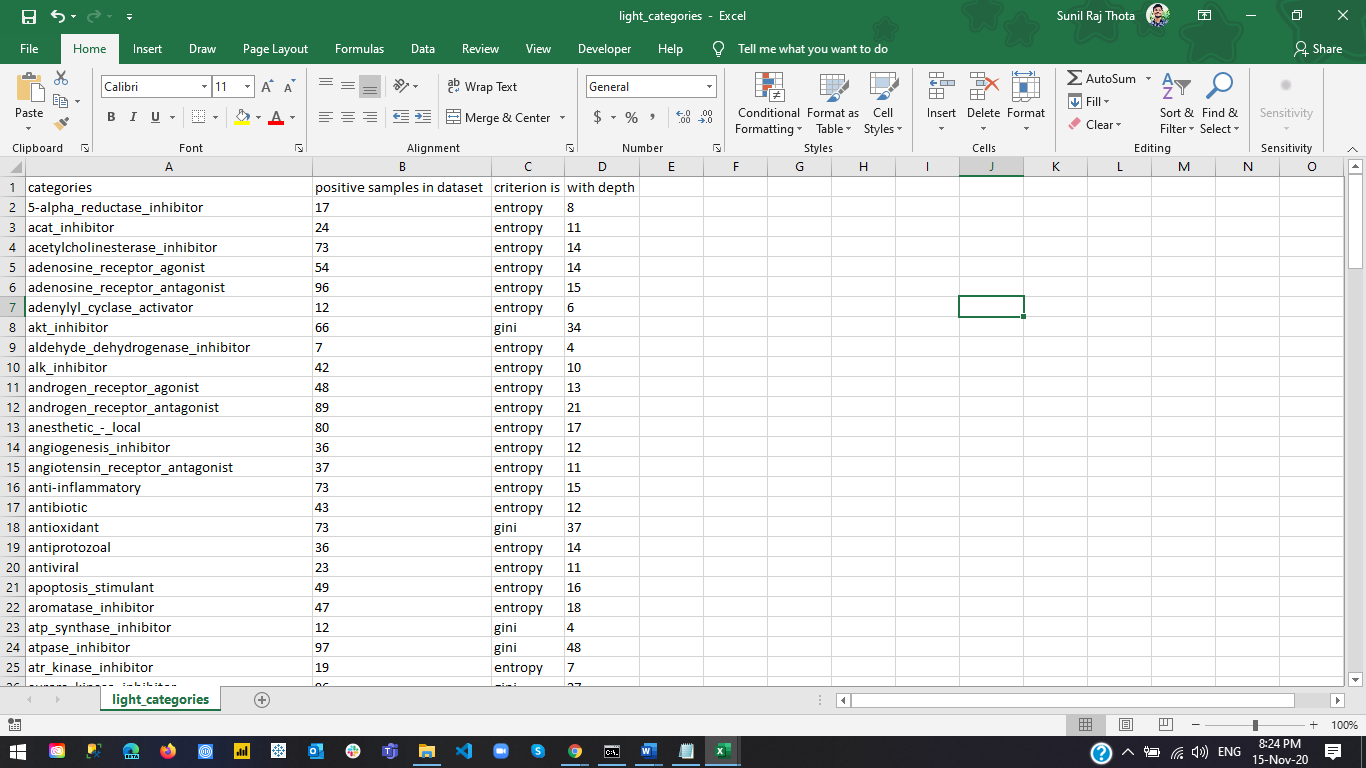
**Disadvantages of Gradient Boosting:**

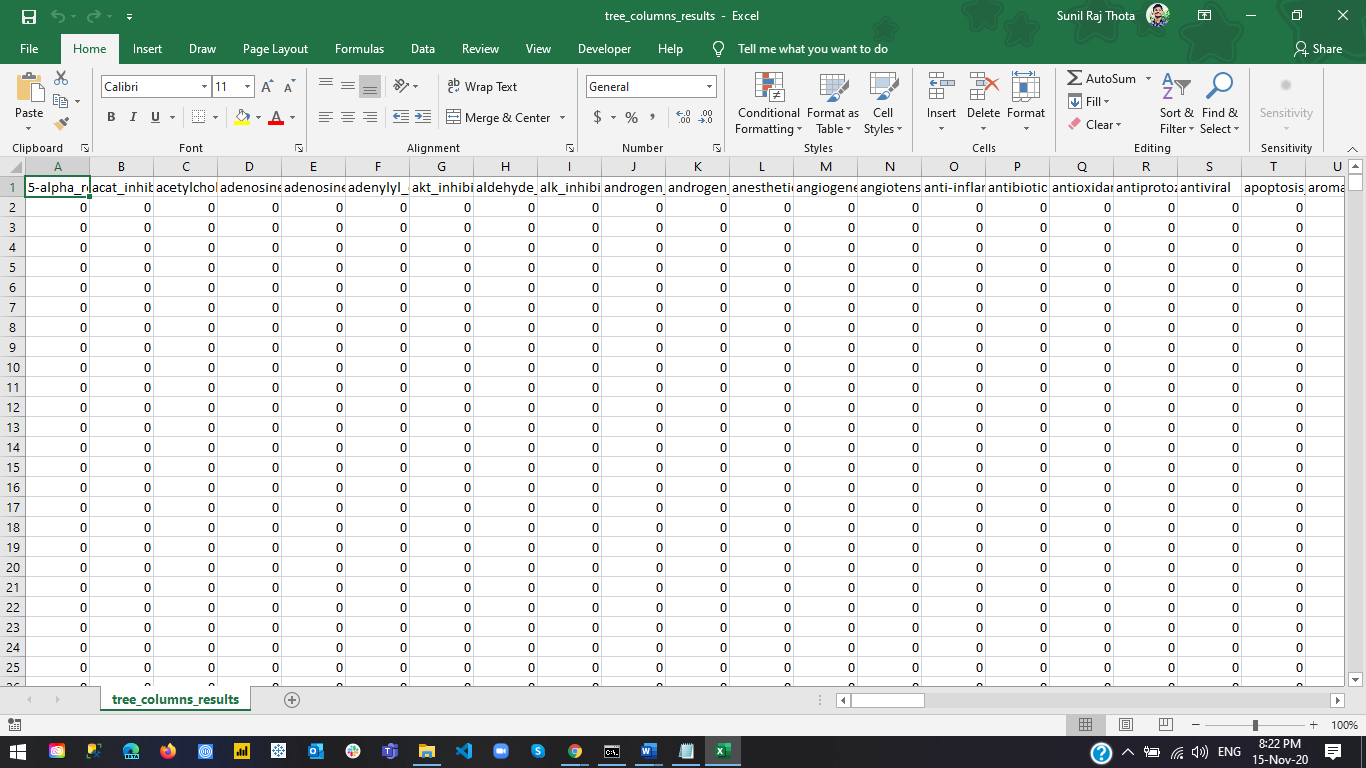
* A small change in the feature set can create radical changes in the model
* Uneasy to understand the predictions

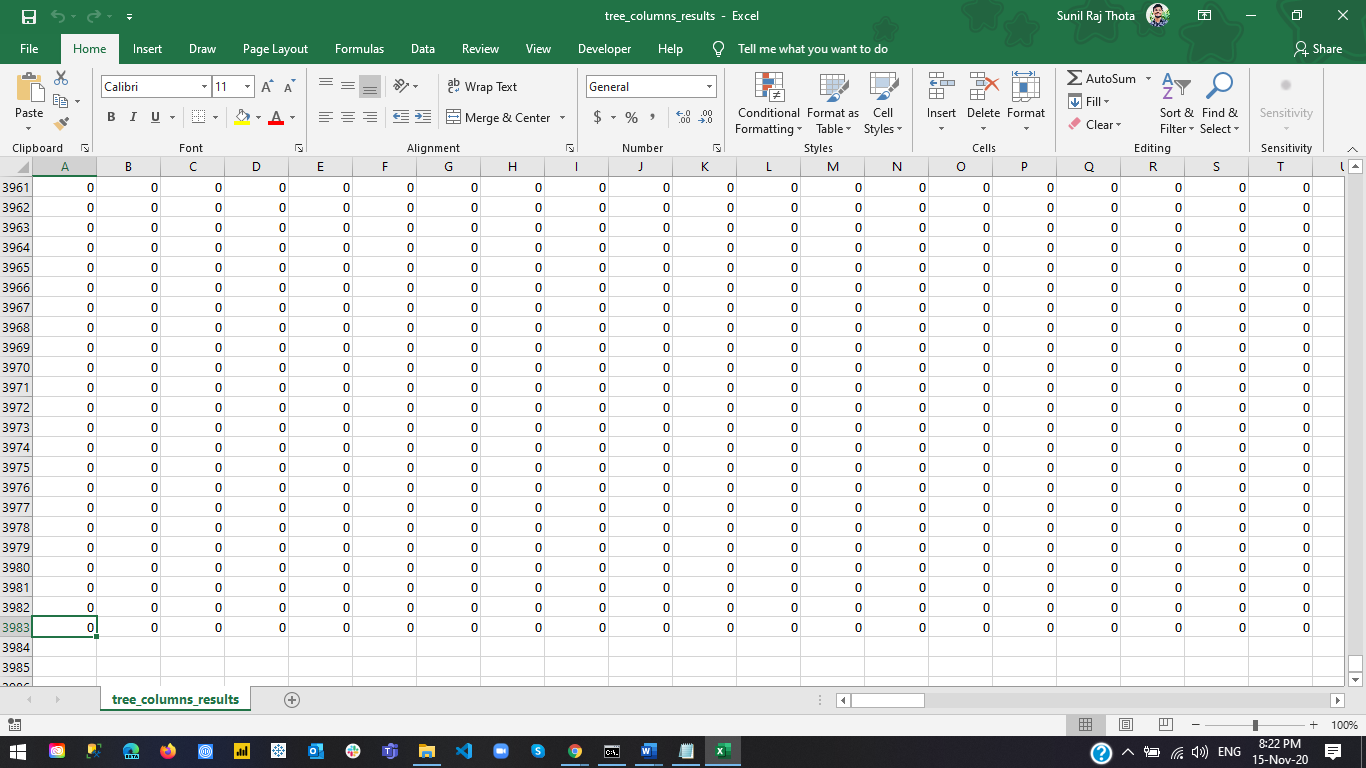
**CONCLUSION**

Following is the calculated predicted values using the above models

It has 3982 rows and 129 columns





From the above values we can see the predicted value from the data set and hence we built various models to achieve

**REFERENCES**

1. Jake, Hoare, (n.d), Gradient Boosting Explained – The Coolest Kid on The Machine Learning Block, Retrieved from https://www.displayr.com/gradient-boosting-the-coolest-kid-on-the-machine-learning-block/
2. Aishwarya Singh, (2018), A Comprehensive Guide to Ensemble Learning (with Python codes), Retrieved from https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/
3. Brownlee, J. (2020, August 14). How to Configure the Gradient Boosting Algorithm. Retrieved October 06, 2020, from https://machinelearningmastery.com/configure-gradient-boosting-algorithm/
4. Dawson, C. (2019, September 26). Gradient Boosting in Python from Scratch. Retrieved October 06, 2020, from https://towardsdatascience.com/gradient-boosting-in-python-from-scratch-4a3d9077367
5. Jorge Leonel (Mar 31, 2019), Bias / Variance in Machine Learning, https://medium.com/@jorgesleonel/bias-variance-in-machine-learning-656a1b58e1c9
6. Megha Mishra (May 26, 2018), REGULARIZATION: An important concept in Machine Learning, https://towardsdatascience.com/regularization-an-important-concept-in-machine-learning-5891628907ea
7. Will Koehrsen (Jan 27, 2018), Overfitting vs. Underfitting: A Conceptual Explanation, https://towardsdatascience.com/overfitting-vs-underfitting-a-conceptual-explanation-d94ee20ca7f9