# Precipitation Prediction using ML

## 1. Walkthrough:

**Week 1**

Setting up the Project:

Download Anaconda and Jupyter on your device.

<https://www.youtube.com/watch?v=uOwCiZKj2rg&feature=youtu.be>

<https://www.youtube.com/watch?v=fiQTb7-rCPo&feature=youtu.be>

Python: <https://www.youtube.com/watch?v=eWRfhZUzrAc>

Pandas: <https://www.youtube.com/watch?v=vmEHCJofslg>

Matplotlib: <https://www.youtube.com/watch?v=yZTBMMdPOww&feature=youtu.be>

Seaborn: <https://www.youtube.com/watch?v=6GUZXDef2U0>

Scikit-learn - <https://www.youtube.com/watch?v=pqNCD_5r0IU>

**Week 2**

Data Importing and Exploration

Use the given Dataset

Dataset: <https://drive.google.com/file/d/1xaspu6UgMI0mBZsOmkiVMIkBQP8V1Jvg/view>

We will use pandas framework to import the data and perform analysis on it.

Precipitation column in the data frame will be our target feature in this model. We have to replace all values **greater than** **0** as **1 (representing precipitation will occur)**, and values that are **equal to 0 representing precipitation will not occur.**

**Week 3**

Handling class imbalance and missing values

* In our dataset, there is an imbalance between examples of where precipitation occurs or not. Use **Matplotlib/Seaborn** to visualize it.
* Most of the ML algorithms used for classification were designed with the assumption of an equal no. of examples in each case. Therefore we need to balance it. The imbalance has to be removed or reduced.
* We will now overbalance the minority class using **sklearn.utils.resample**.
* **Use** [**this.**](https://scikit-learn.org/stable/modules/generated/sklearn.utils.resample.html)
* We will now check for **null values**. (remember **Pandas?**)
* If any feature contains many null values, we will drop it.
* Now, we will convert the rest of the null values with mode.

**Standardizing data and feature selection**

* Feature selection will be made using the **chi-square test.**

What is the chi-square test for feature selection?

[**Read this**](https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223)**.**

How will we do this?

Use [**SelectKBest**](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.SelectKBest) and [**chi2**](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html#sklearn.feature_selection.chi2)**.**

* We will now **normalize** our data.(Revise Pandas)

<https://www.geeksforgeeks.org/data-normalization-with-pandas/>

**Finally,**

So guys we are in the final phase of our project.

**Training model using different techniques**

* Split data into test and train datasets.
* We can use logistic regression classifier, decision tree classifier, neural networks, etc on training dataset.
* Calculate accuracy, precision, recall, F-1 score, and ROC\_AUC on the test dataset and visualize it.
* Plot confusion matrix using sklearn.

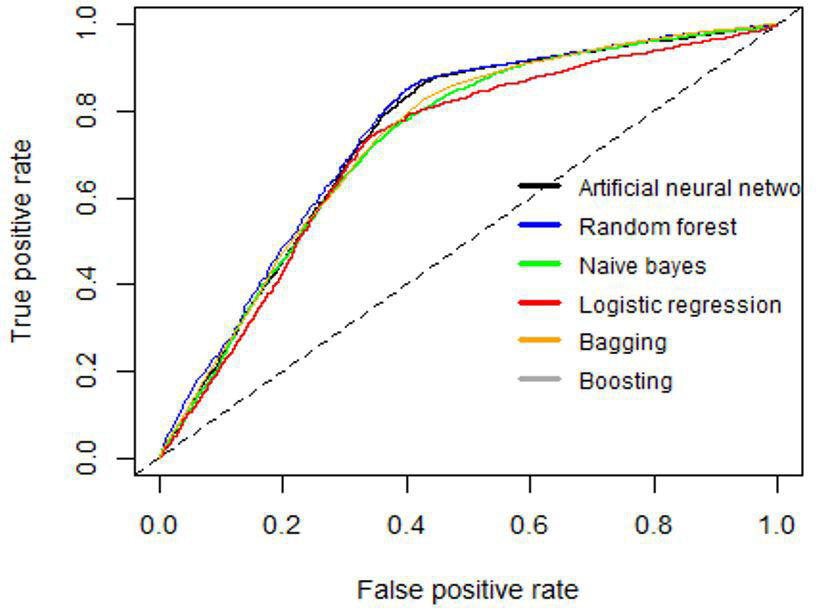
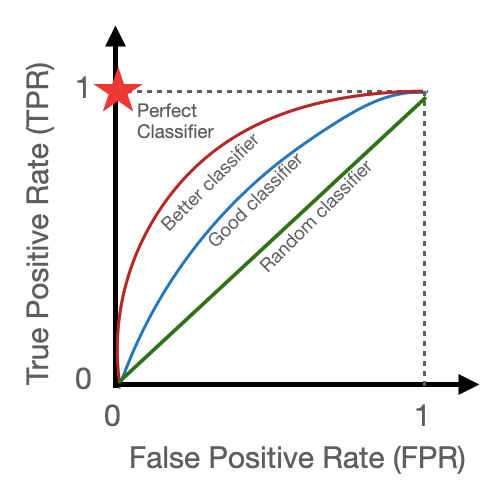
**Model Comparison**

Compare models based on accuracy and ROC\_AUC score and visualize it using seaborn

## 2. Chi-square technique

1. The idea is to find out features that aren’t useful
2. In simple words, higher the Chi-Square value the feature is more dependent on the response(y) and it can be selected for model training.
3. Higher the p-value, more is the variables independent of the response(y) and can not be considered for model training
4. from sklearn.feature\_selection import chi2
5. chi\_scores = chi2(X,y)
6. chi\_scores
7. # here first array represents chi square values and second array represents p-values
8. p\_values = pd.Series(chi\_scores[1],index = X.columns)
9. p\_values.sort\_values(ascending = False , inplace = True)
10. p\_values.plot.bar()

## 3. ROC/AUC

1. An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate.
2. 
3. AUC ROC stands for “Area Under the Curve” of the “Receiver Operating Characteristic” curve
4. 
5. from sklearn.metrics import roc\_curve, auc
6. fpr\_LR, tpr\_LR, \_ = roc\_curve(y\_test, y\_predict\_LR)
7. roc\_auc\_LR = auc(fpr\_LR, tpr\_LR)
8. fpr\_RF, tpr\_RF, \_ = roc\_curve(y\_test, y\_predict\_RF)
9. roc\_auc\_RF = auc(fpr\_RF, tpr\_RF)
10. fpr\_XGB, tpr\_XGB, \_ = roc\_curve(y\_test, y\_predict\_XGB)
11. roc\_auc\_XGB = auc(fpr\_XGB, tpr\_XGB)
12. plt.figure()
13. plt.plot(fpr\_LR, tpr\_LR, label='Logistic Regression (AUC = %0.2f)' % roc\_auc\_LR)
14. plt.plot(fpr\_RF, tpr\_RF, label='Random Forest (AUC = %0.2f)' % roc\_auc\_RF)
15. plt.plot(fpr\_XGB, tpr\_XGB, label='XG BOOST (AUC = %0.2f)' % roc\_auc\_XGB)
16. plt.plot([0, 1], [0, 1], 'k--')
17. plt.xlim([0.0, 1.0])
18. plt.ylim([0.0, 1.05])
19. plt.xlabel('False Positive Rate')
20. plt.ylabel('True Positive Rate')
21. plt.title('Receiver Operating Characteristic')
22. plt.legend(loc="lower right")
23. plt.show()