Gina Jozef
4/19/22

ATOC5860 – Application Lab #6 Machine Learning with Weather Data Spring 2022

Note: You will need to use the python environment provided (environment.yml), especially for notebook #2. These notebooks were written by Eleanor Middlemas in 2020 (https://github.com/e-middlemas/ML_application_lab). They were last adapted/updated for use in ATOC5860 during Spring 2022.

Notebook #1
ATOC5860_applicationlab6_cluster_mesa_data.ipynb

LEARNING GOALS
1) Use k-means clustering as an example of unsupervised (grouping events into different categories) machine learning
2) Become familiar with the limits and applicability of K-means clustering to detect seasons in Boulder, Colorado
3) Assess sensitivity of K-means to standardization, changing the variables used for the clustering (also called "features"), and number of clusters (4 vs. 3).

DATA and UNDERLYING SCIENCE:
You will be working with weather data from the NCAR Mesa Laboratory in Boulder, Colorado. We'll call this dataset the "Mesa dataset". The data go from 2016-2021. Information on the site and the instruments is here: https://www.eol.ucar.edu/content/ncar-foothills-lab-weather-station-information. Real-time data from the site is here: https://archive.eol.ucar.edu/cgi-bin/weather.cgi?site=ml. Note: Each year in this dataset has 365 days. Leap year data (i.e., Feb. 29 data for 2016 and 2020 have been excluded.)

In this notebook, you use K-means clustering to classify the mesa dataset weather data into different clusters. Why would we cluster weather observations? We already know which observations are in which season by looking at the date. But we all know that a day in February sometimes feels like summer and a day in September can feel like winter. We often have multiple seasons in a single week... So this could be quite fun to see how the algorithm decides how to cluster our data and assign each day to a "season". :) Will each cluster will look like a season – On Va Voir (We'll See)!

Questions to guide your analysis of Notebook #1:

1) **Start with 4 clusters. Cluster the data at 17 UTC (mid-day in Colorado). What is the seasonal occurrence of the 4 clusters? Do the 4 clusters correspond to Fall, Winter, Spring, and Summer? Why or why not?**

Using four clusters, the only season that stands out is Summer, which is evident by the fact that cluster 2 shown in Figure 1 below has a high concentration of datapoints in Summer, but few datapoints in the other times of year. However, for all of the other clusters, the datapoints are more evenly spaced throughout the year, aside from a low concentration in summer.

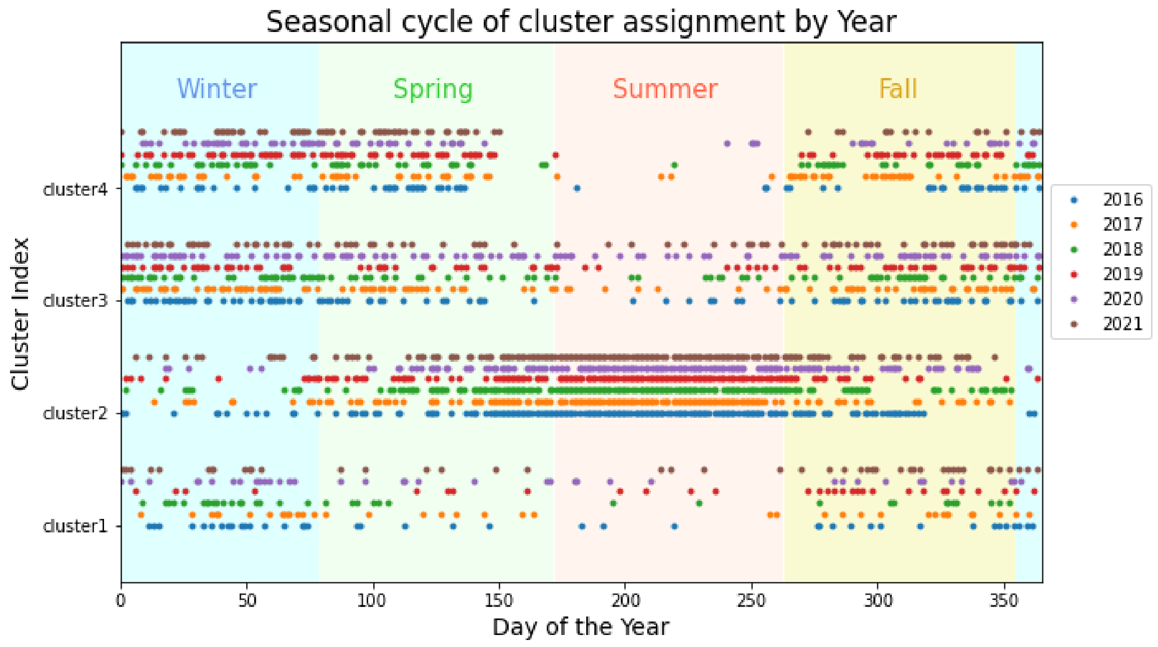


Figure 1: Cluster index versus time of year (using four clusters), with shading to indicate what we define to be Winter, Spring, Summer and Fall.

2) Based on 2D and 3D scatter plots of the cluster centers and the data – Which weather variables help (or NOT help) define the clusters?

The plot of wind speed versus temperature shown in Figure 2 below is helpful to define the clusters, as it differentiates at least three clusters from each other (yellow, red, blue) however the last cluster (grey) is kind of spread out among the rest of the clusters. The plot of relative humidity versus wind direction shown in Figure 3 below is sort of helpful, as three clusters (again yellow, red, blue) are essentially in the four corners of the plot, but the last cluster (grey) is on top of the red cluster. The 3D plot of relative humidity versus wind direction versus temperature in Figure 4 below does not provide a ton of value. While the clusters are separated more in some parts, they also overlap quite a bit throughout the plot.

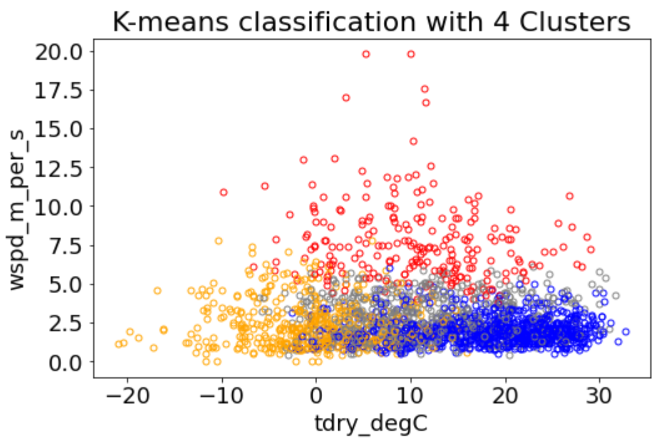


Figure 2: Wind speed versus temperature, color coded by the four clusters.

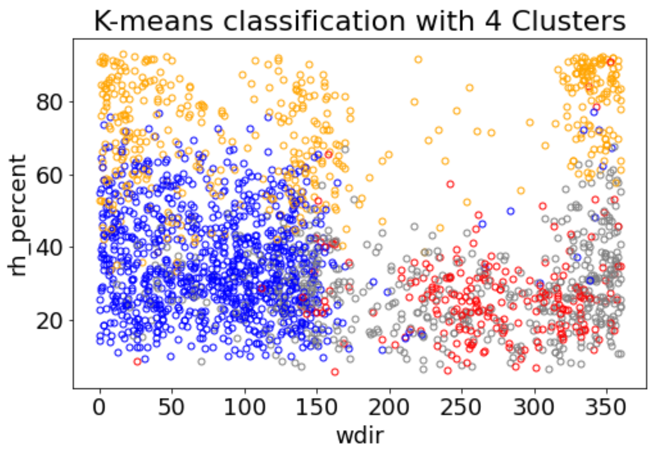


Figure 3: Relative humidity versus wind direction, color coded by the four clusters.

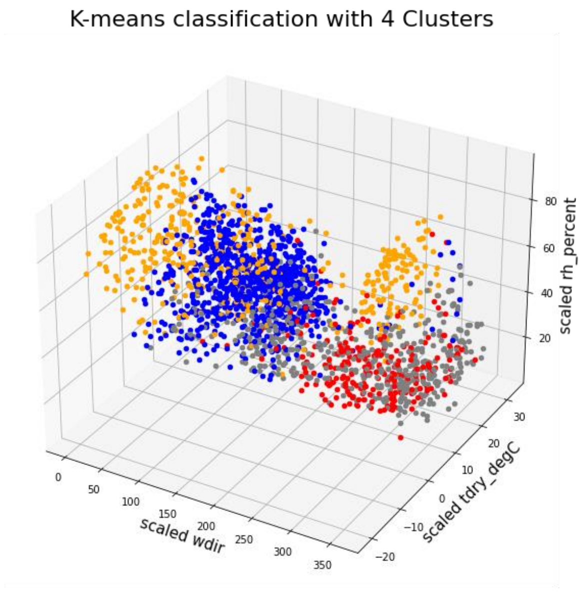


Figure 4: Relative humidity versus temperature versus wind direction, color coded by the four clusters.

Gina Jozef
4/19/22

3) **What do the clusters show during the time period from September 5-15, 2020 (Labor Day 2020)? Are the cluster assignments consistent with the weather experienced over that time period? Are there other date ranges that you would like to check out?**

The clusters put the data during this time period in the cluster that most closely aligns with winter, where all of the surrounding datapoints are in the summer cluster. This is consistent with the weather experienced over that time period, since it snowed, but the days before and after were hot, like a typical summer day.

4) **Re-run the analysis. But now use three clusters instead of four clusters. Compare your cluster analyses for 4 clusters and 3 clusters. Do the results for 4 clusters or 3 clusters make more sense to you based on your analysis and also your experience living in Boulder, Colorado? Which number of clusters provides a better fit to the data?**

Figures 5-8 below show the same results when using 3 clusters instead of 4. Looking at Figure 5, it appears that clusters 2 and 3 are distinct and represent not-summer and summer respectively. Then, cluster 1 is more spread out amongst all seasons, but with fewer datapoints in the summer. Thus, while this is perhaps an improvement from using 4 clusters, it still isn't perfect. Looking at wind speed versus temperature, shown in Figure 6, when using 3 clusters then looking at these two variables is helpful and is differentiable between the three clusters. When looking at relative humidity and wind direction, shown in Figure 7, two of the clusters are differentiable (grey and red) but the third cluster (blue) is more spread out. In the 3D plot shown in Figure 8, the red and blue clusters are more differentiable, but the grey variable is mixed throughout the other two. But all in all, the different clusters are more differentiable when 3 clusters are used rather than 4, so 3 clusters likely provide a better fit to the data.

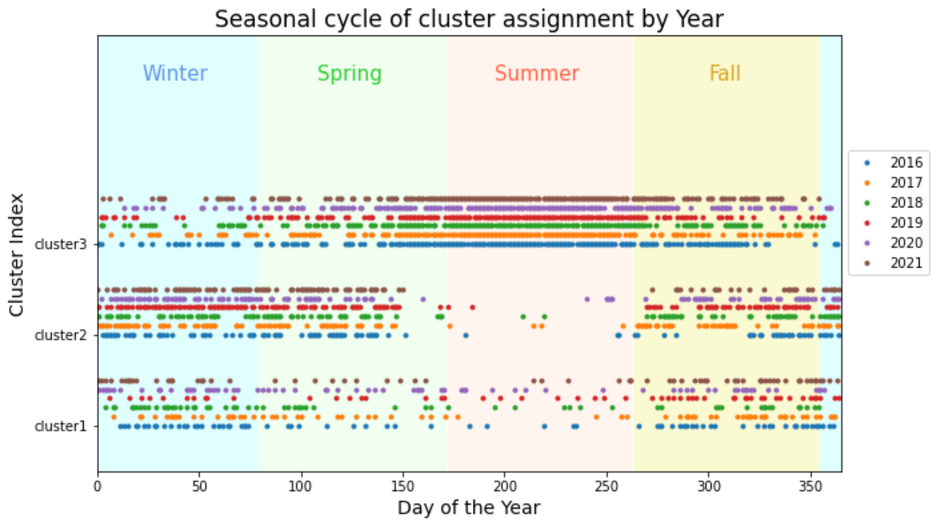


Figure 5: Cluster index versus time of year (using three clusters), with shading to indicate what we define to be Winter, Spring, Summer and Fall.

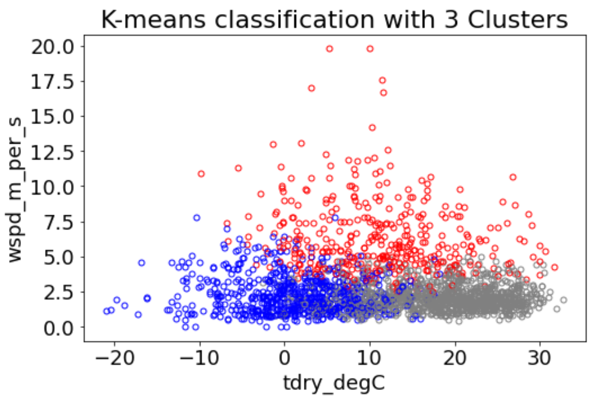


Figure 6: Wind speed versus temperature, color coded by the three clusters.

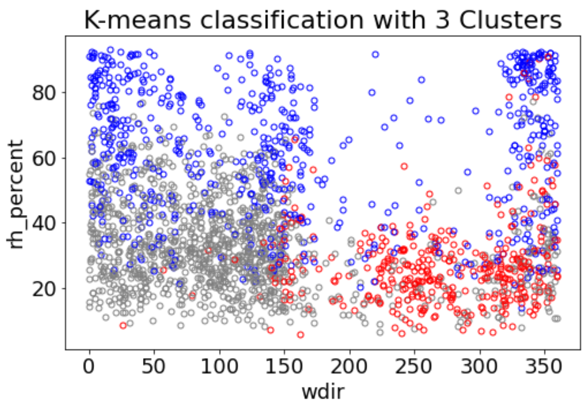


Figure 7: Relative humidity versus wind direction, color coded by the three clusters.

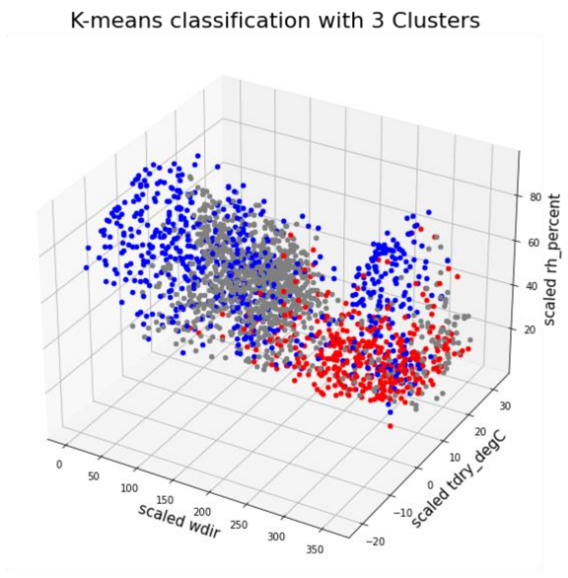


Figure 8: Relative humidity versus temperature versus wind direction, color coded by the three clusters.

Out of curiosity, I also tried clustering with only 2 clusters. Based on Figure 9 below, this seems to be worse than using 3 clusters, since the two clusters are more difficult to differentiate.

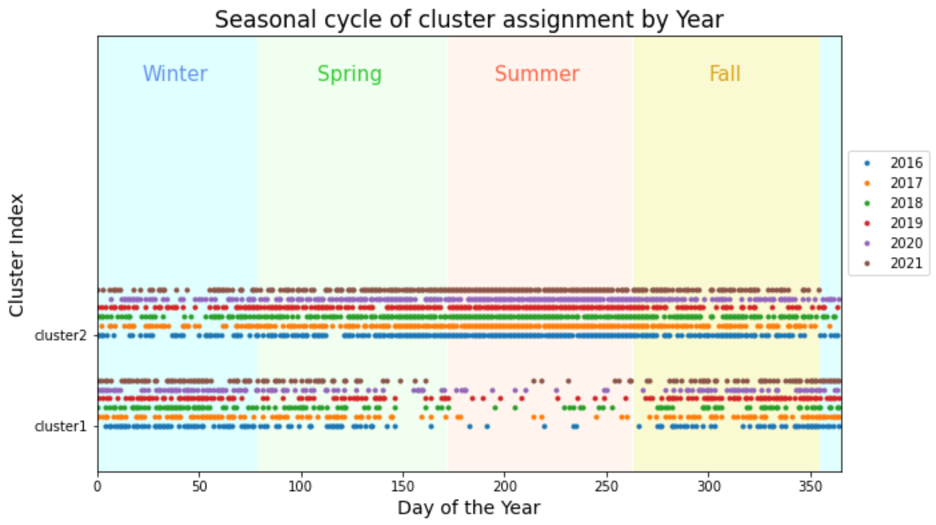


Figure 9: Cluster index versus time of year (using two clusters), with shading to indicate what we define to be Winter, Spring, Summer and Fall.

Notebook #2

OPTION #1: ATOC5860_applicationlab6_supervised_ML.ipynb – Use environment.yml

or

OPTION #2: supervised.ipynb – Run in Google CoLabs

Note: You will need to change the google drive paths to match those on your computer.

LEARNING GOALS:

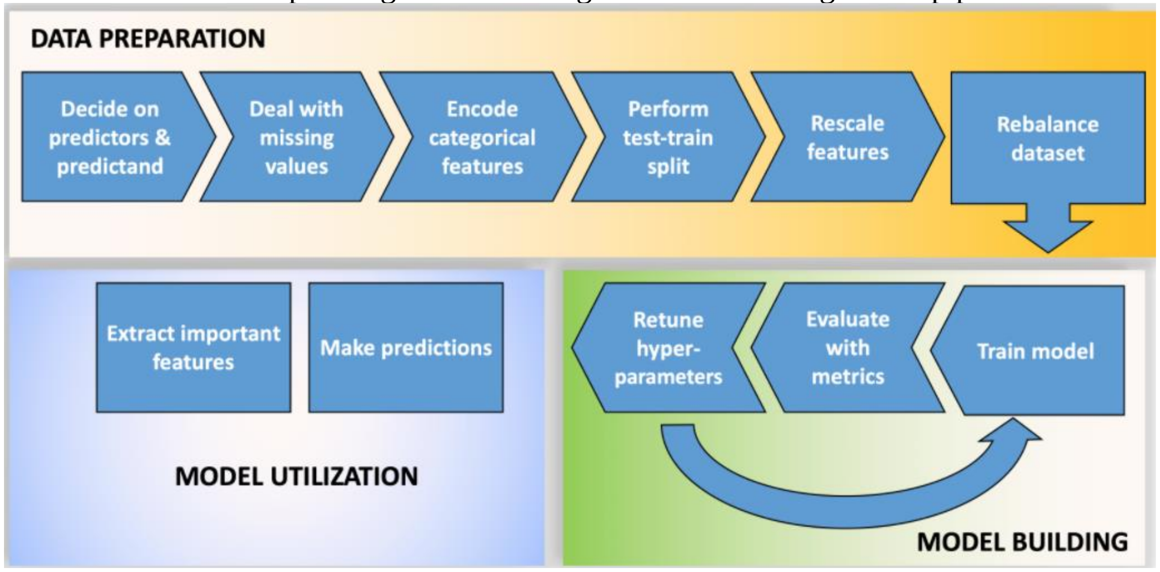
1) See an example of the data processing pipeline (workflow) required to utilize supervised machine learning techniques.
2) Implement and compare four different supervised learning algorithms
3) Understanding two outcomes of supervised learning algorithms: prediction and feature importance.
4) Start building a foundation for future machine learning including the following terms: cross-validation, training vs. testing data, metrics (accuracy, recall, precision, f1 score, etc.), overfitting/underfitting, balancing datasets, hyperparameters, & feature importance. Some future learning resources are provided… but there's a lot available! *Share resources you find valuable.*

DATA and UNDERLYING SCIENCE:

We will use the Christman dataset which contains weather observations from Fort Collins, Colorado for the year 2016. We will build and train four machine learning models to predict something we already know from the dataset: ***Is it raining?***. The point is not to conduct cutting-edge research or make novel predictions. Instead, the purpose here is to showcase supervised machine learning (ML) models and methods. By the end, we hope you can walk away with more confidence to learn and apply these tools to new problems.

Let's say you want to determine which features or atmospheric variables are the best predictors of rainfall. Often, one simply regresses some metric of precipitation onto various atmospheric variables. Then, you assume that whatever returns the highest regression coefficient is the best predictor. While this approach with linear regression presents a fine first guess, it poses a few problems. Linear regression assumes: 1) atmospheric variables are linearly related to precipitation, 2) atmospheric variables are uncorrelated. Yet, both are false assumptions. While a linear relationship between predictor & predictand is a good first guess, why limit yourself to linearity when you can just as easily relax that assumption using supervised Machine Learning...

This notebook will step through the following Machine Learning model pipeline:



After prepping the data, we will build and train four machine learning models and make predictions with them. The four machine learning models we will implement are: Logistic regression, Random Forest, Singular vector machines/classifier, Neural Network. Finally, we will determine which variable ("feature") is the best predictor, i.e., we will assess "feature importance".

Pros/Cons of these Methods (from Eleanor Middlemas)

1. Logistic regression tends to overgeneralize or underfit data, but is easy to implement, to understand and easy to back out feature importance.
2. Singular Vector Machines are great at capturing complex relationships, but cannot back out feature importance. Plus, the use of the kernel makes them hard to interpret.
3. Random forests are easier to understand, generally do not overfit, and can capture complex relationships, and can provide feature importance, but they can be slow to train and there are a lot of hyperparameters to choose from.

Gina Jozef
4/19/22

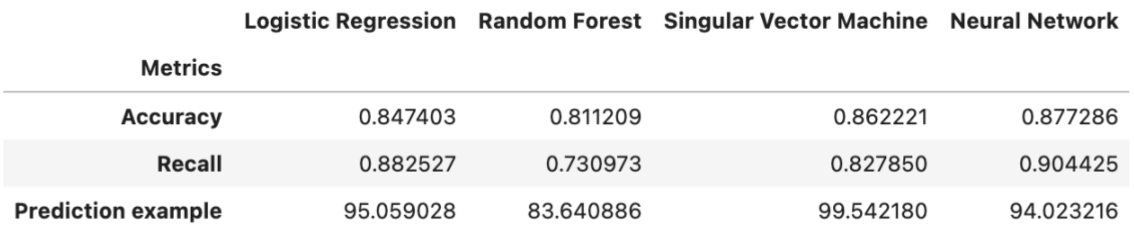
4. Neural Networks are great at capturing complex relationships. But they are slow to train and are susceptible to overfitting.

Questions to guide your analysis of Notebook #2 – See also questions at the end of supervised.ipynb:

1) Which machine learning model performs the best to predict rainfall? What metrics did you use to make this assessment?

The singular vector machine (SVM) performs best to predict rainfall. One metric I am using to determine this is the percent change that it is raining for the example day chosen. The SVM model says there is a 99.5% chance that it is raining on the day that we know it rained, based on the parameters we give the model. The next highest percent predicted is 95% from the logistic regression. All numbers are provided in Table 1 below. The SVM also had a relatively high accuracy, though the accuracy of the neural network is higher. The SVM method also had fewer instances of a false negative than the random forest, but more instances of a false negative than the neural network (Figure 1). The SVM also had the least instances of a false positive of any method (Figure 1). However, the SVM model as the second to lowest recall of all the methods. Nonetheless, considering all factors, I think the SVM model performed best.

Table 1: Accuracy, recall, and predicted precipitation probability for all of the machine learning models.



Metrics	Logistic Regression	Random Forest	Singular Vector Machine	Neural Network
Accuracy	0.847403	0.811209	0.862221	0.877286
Recall	0.882527	0.730973	0.827850	0.904425
Prediction example	95.059028	83.640886	99.542180	94.023216

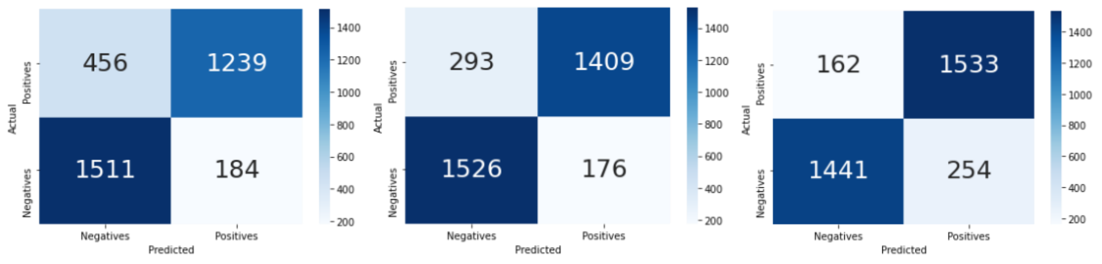


Figure 1: From left to right, confusion matrix for the output from the random forest, singular vector machine, and neural network.

2) Describe the difference between accuracy and recall. Why did we choose to use accuracy, recall, and predicted precipitation probability as a way to compare models? In forecasting: when is a false positive (you said it would rain, it didn't rain) preferred over a false negative (you said it wouldn't rain, it did rain)?

Gina Jozef
4/19/22

Accuracy = (TP + TN)/(total): The proportion of precipitating hours or non-precipitating hours that are correctly predicted by the model.

Recall = TP/(TP + FN): The proportion of precipitating hours that are correctly predicted by the model.

We chose to use the accuracy, recall, and predicted probability to compare models because these three metrics cover the range of the options in terms of how the model is able to predict what occurs, including both if the model can predict whether it is precipitating or not, as well as how well the model is able to predict if there is precipitation. In forecasting, a false positive is generally preferred over a false negative because with a false positive, people can still prepare for the weather event, and it is only a mild frustration if that weather doesn't end up occurring. But with a false negative, people may not prepare for a weather event and could then be in a bad situation if it ends up occurring. This is especially true for extreme weather events in which an evacuation may be necessary.

3) One important "gotcha" in a machine learning workflow or pipeline is the order of data preparation. **Why should one should perform the train-test split before feature scaling and rebalancing?** *Hint: think about using a trained model for future predictions*. Do you want your scaling of the testing data to depend on the training data? Why perform a test-train split at all?

You want to perform a test-train split before feature scaling because the number of instances of a certain feature will be different depending on the way you split your data into the training set and the testing set. Doing a test train split is an important step because it allows you to evaluate a model's performance with the holdout testing data. Applying the test-train split must be performed before each time the model is trained to ensure you are not baking in any bias among the models you train. You don't want the scaling of the testing data to depend on the training data because you want the two sets to be as independent of each other as possible, so you can test the efficacy of the model on the test samples. If you scale the testing set based on the training data, this may bias your results.

4) Collinearity, or non-zero correlation among features, results in a model that is overly complex, reduces the statistical significance of the fit of the model, and prevents one from correctly identifying the importance of features. ***Are there features included in our machine learning models to predict rain in the Christman dataset that are collinear?*** If so, how do you think we should address this collinearity? A couple of suggestions: If we don't have that many features, we could use our meteorological expertise to simply remove one of the features that shares collinearity with other features. Another way to address collinearity is to use feature regularization, or add weights that penalize features that add noise, ultimately reducing model complexity.

Some features in our dataset that may be collinear are RH versus temperature and dewpoint temperature. Since we include all of these factors, it appears that temperature has a higher importance in predicting precipitation than it actually does, since temperature is a component of calculating RH. Also, there is collinearity between temperature and pressure, so again, providing both is giving some of the same information. However, pressure and RH do not have collinearity, so these two variables provide unique information. To account for this collinearity, it may be best to simply remove temperature and dewpoint temperature from the variables in consideration.