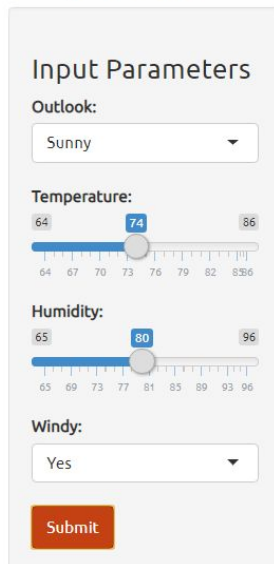


Building a ML / Data-Driven Web Application in R

- Again using the Shiny web framework
- This will make use of the random forest algorithm. It aims to predict whether or not to play golf as a function of the input weather parameters.
- Recall def'n of **parameter** in stats terms: sigma or p; it's a value that tells you something about a population ((weather outlook, temperature, humidity and wind))
 - Difference between a stat and a parameter?
 - Both are similar since they are both descriptions of groups, however, stats describe a sample whereas a parameter describes the WHOLE pop'n
- This web app will be based on the Weather dataset from weka data mining software
- Link: <https://github.com/dataprofessor/data/blob/master/weather-weka.csv>
- Note: install libraries you don't have; i'm missing Rcurl and randomForest, hence the code below
- `install.packages(c("randomForest", "RCurl"))`
- Looking at the data, the 4 variables are outlook, temperature, humidity, windy
 - The class label is play (whether or not to play golf based on variables)
 - This is a function of the weather conditions; i.e. whether it is sunny, temp, humidity is high or low or medium, whether it is windy; true or false etc.

Play Golf?



Status/Output

[1] "Calculation complete."

Prediction	no	yes
yes	0.30	0.70

- with all variable factors being taken into account, no has a 30% probability while yes has 70% prob.
 - Hence, the overall decision / prediction being stated as yes

Play Golf?

Input Parameters

Outlook:

Temperature:

Humidity:

Windy:

Status/Output

[1] "Calculation complete."

Prediction	no	yes
no	0.84	0.16

- decision would obv. Be no
- with hot and humid temperatures, the

What does the data set actually look like?

- **view(weather)**

	outlook	temperature	humidity	windy	play
1	sunny	85	85	FALSE	no
2	sunny	80	90	TRUE	no
3	overcast	83	86	FALSE	yes
4	rainy	70	96	FALSE	yes
5	rainy	68	80	FALSE	yes
6	rainy	65	70	TRUE	no
7	overcast	64	65	TRUE	yes
8	sunny	72	95	FALSE	no
9	sunny	69	70	FALSE	yes
10	rainy	75	80	FALSE	yes
11	sunny	75	70	TRUE	yes
12	overcast	72	90	TRUE	yes
13	overcast	81	75	FALSE	yes
14	rainy	71	91	TRUE	no

- 5 cols: 4 variables/ parameters, and 1 class label
- Quicker view of some data, first 6 rows; use: **head(weather)**
- Data type of the data set?
 - **str(weather)**

```
> str(weather)
'data.frame': 14 obs. of 5 variables:
 $ outlook : Factor w/ 3 levels "overcast","rainy",...: 3 3 1 2 2 2 1 3 3 2 ...
 $ temperature: int 85 80 83 70 68 65 64 72 69 75 ...
 $ humidity : int 85 90 86 96 80 70 65 95 70 80 ...
 $ windy : logi FALSE TRUE FALSE FALSE TRUE ...
 $ play : Factor w/ 2 levels "no","yes": 1 1 2 2 2 1 2 1 2 2 ...
```

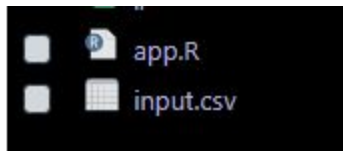
- “Play” is a categorical label

Random Forest Model

- A randomForest model will be created using the four variables comprising of outlook, temp, humidity, windy as the input variable

```
# Build model
model <- randomForest(play ~ ., data = weather, ntree = 500, mtry = 4, importance = TRUE)
```

- - The “play” variable will be used as the output variable OR IOW the variable we want to predict
 - Data is “weather”
 - Number of trees: 500
 - Since there are 4 input variables, will use “mtry = 4”
- Will learn more about the Random Forest Model itself; pretty complex...
- Apply model to make a prediction



- notice the input file; the model will be applied to this

```
121 Output <- data.frame(Prediction=predict(model, test), round(predict(model, test, type="prob"), 3))
122 print(Output)
123
```

- Running the highlighted code will give us an ERROR

```
Error in as.data.frame(newdata) : object 'test' not found
> |
```