Predicting Solar Radiation ☀️

at the HI-SEAS Mars habitat

Natalie Weaver
05 February 2021

The Data

The Data

The source

Collected at the HI-SEAS Mars habitat weather station

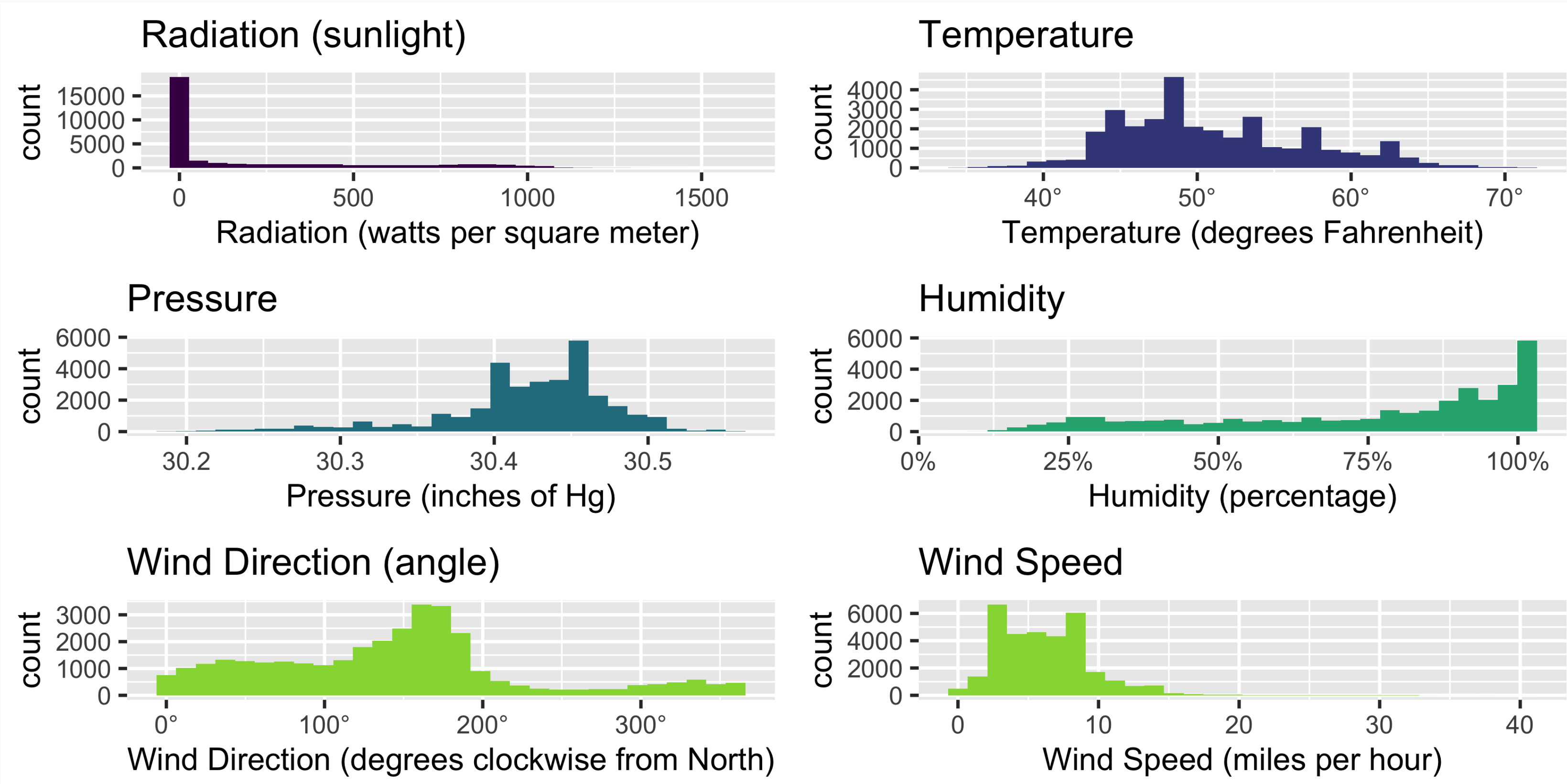
- Habitat used by NASA for human behavior research in conditions simulating a long-term mission to Mars
- Data collected September through December of 2016
- 32,686 observations

Downloaded from Kaggle

- Published by NASA for a hackathon challenge
- Uploaded by user Andrey in 2017

The Data

The variables



Also the date and time of the observation, and the sunrise and sunset times for the date of the observation.

The Data

Feature engineering

`is_daytime` (logical):

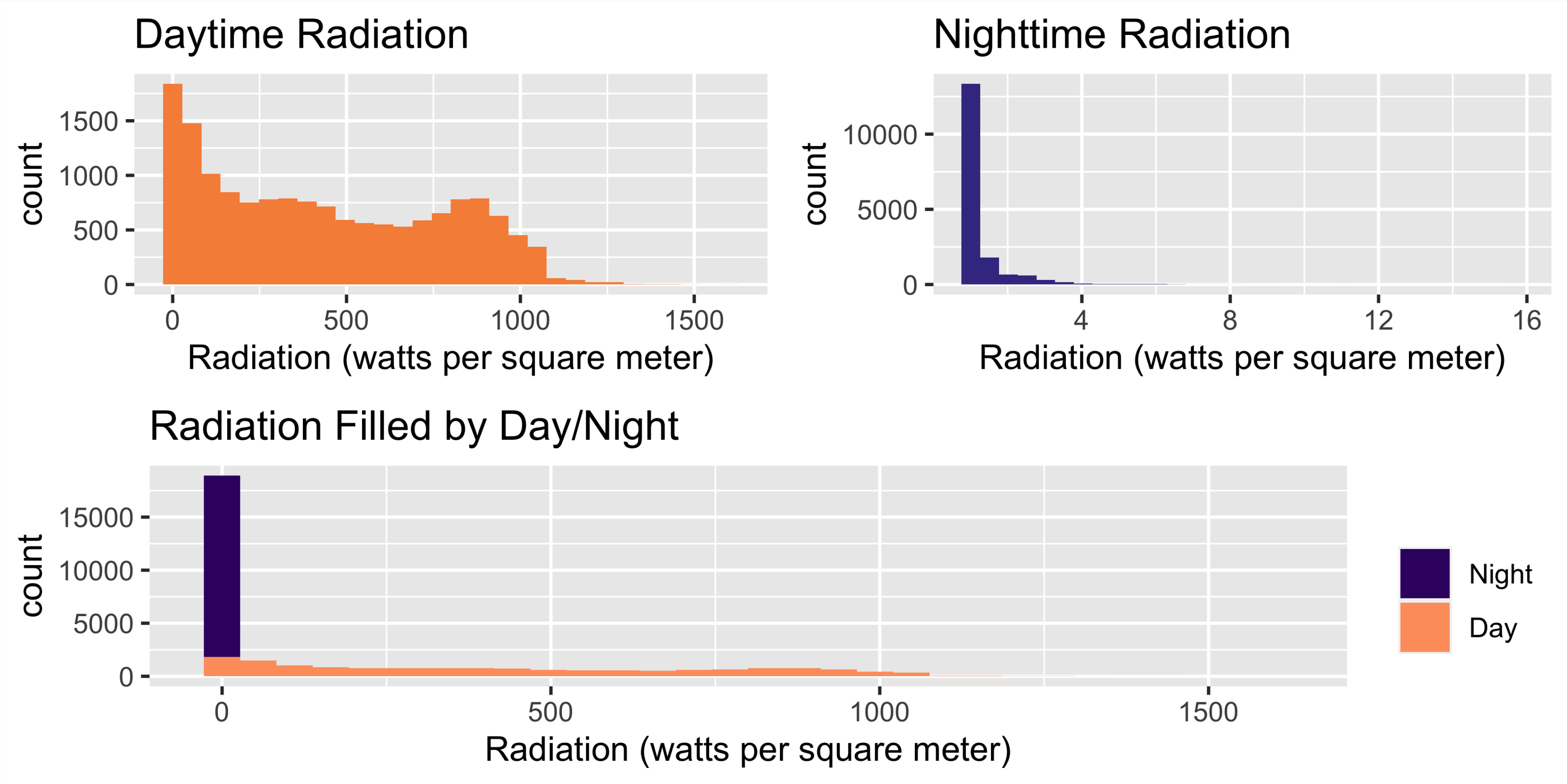
- `0` if an observation occurred after sunrise and before sunset (the daytime)
- `1` otherwise (observation occurred at night).

`wind_direction_factor` (factor)

- `"north"` if wind angle was $< 45°$ or $> 315°$ (i.e. within $45°$ of due North)
- `"east"` if wind angle was $> 45°$ and $< 135°$
- `"south"` if wind angle was $> 135°$ and $< 225°$
- `"west"` if wind angle was $> 225°$ and $< 315°$

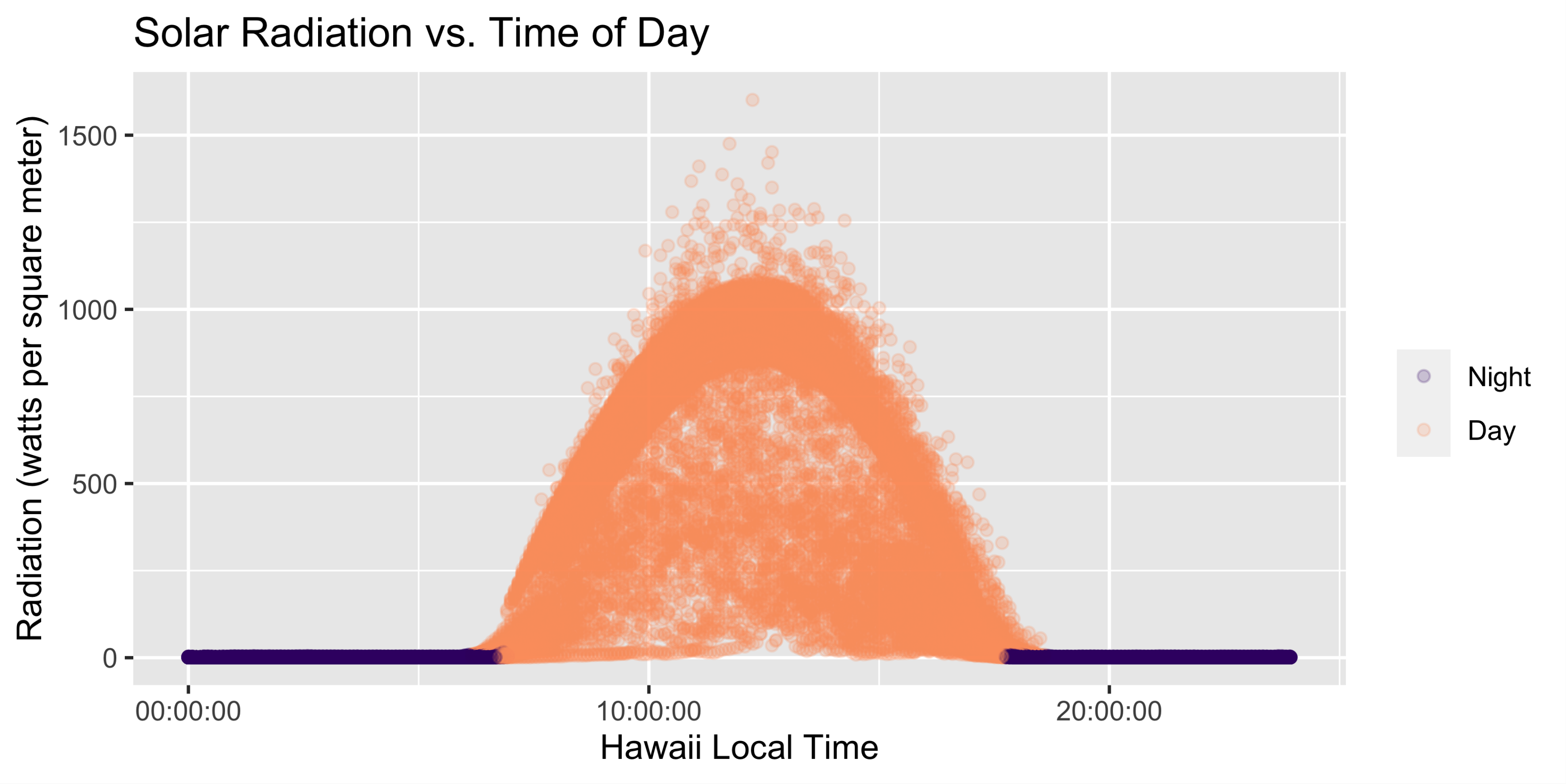
The Data

Updated radiation histograms



The Data

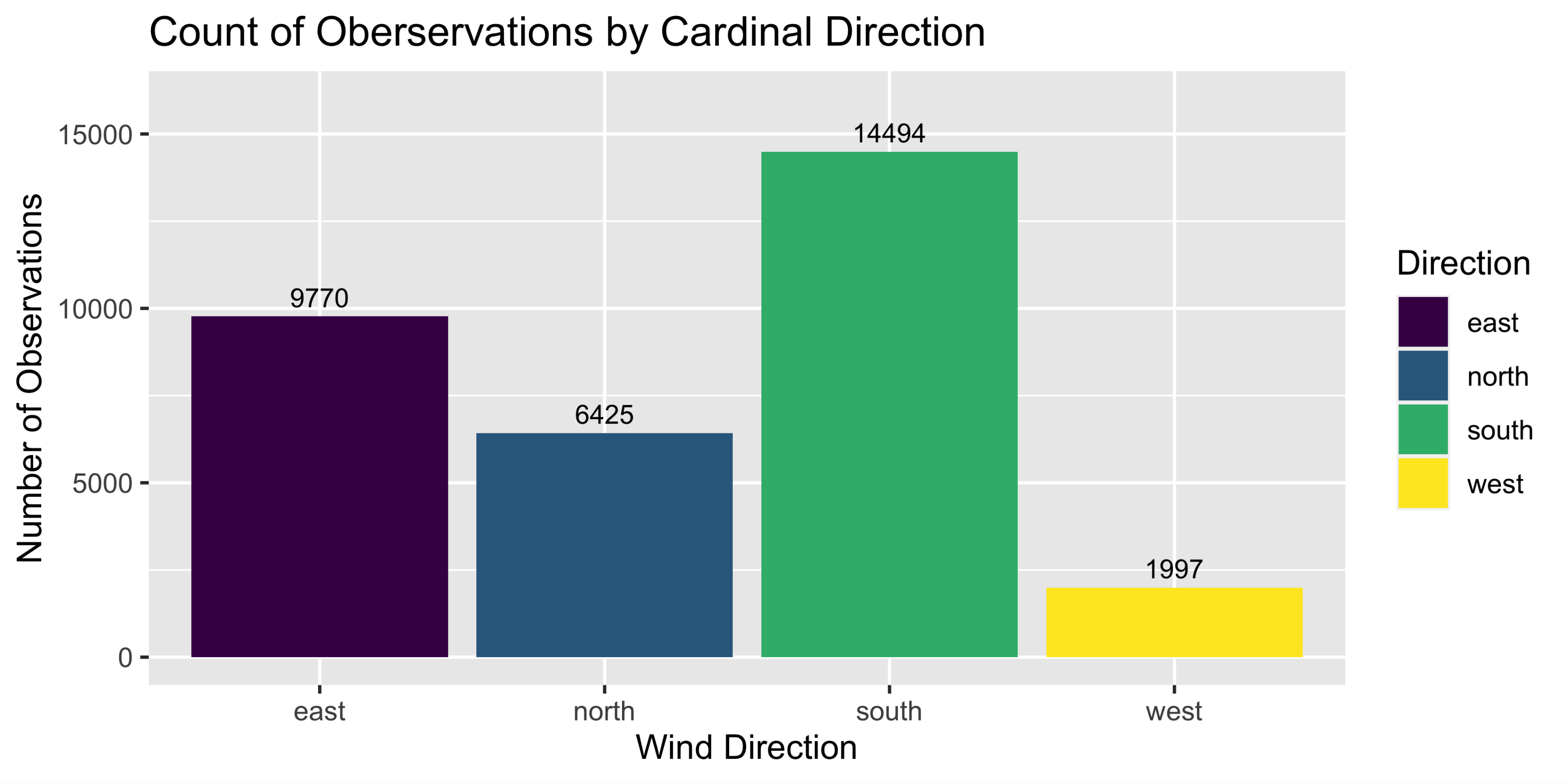
Scatterplot of radiation against time of day



Most of the near-0 daytime radiation values were observed at dawn and dusk.

The Data

Bar chart of wind direction



Prediction Models

Prediction Models

The plan of attack

Try to predict the level of solar radiation using three machine learning methods:

- Penalized regression (elasticnet)
- K-Nearest Neighbors
- Tree-based methods (decision trees, random forest)

Evaluate the models by calculating the RMSE of their predictions on held-out test data.

Prediction Models

Data preparation

Before we can fit these models, we need to prepare the data:

- Add an `id` column to number the rows, easier to keep track of
- Throw out `wind-direction-degrees` and `unix-time`
- Convert all dates and times to doubles so they play nice with model fitting functions
- Create a standardized version of the data for KNN and elasticnet models (and keep the original data to use for tree-based models)
- Hold out 20% of the data to evaluate model performance at the very end.

Prediction Models

Elasticnet model

How it works:

- Linear combination of Ridge and LASSO regressions
  - Ridge: OLS with shrinkage penalty equal to sum of squared coefficients
  - LASSO: OLS with shrinkage penalty equal to sum of absolute coefficients

Parameters to tune:

- `λ`: scalar for the shrinkage penalty
- `α`: balance between Ridge and LASSO
  - `0` = 100% Ridge
  - `1` = 100% LASSO

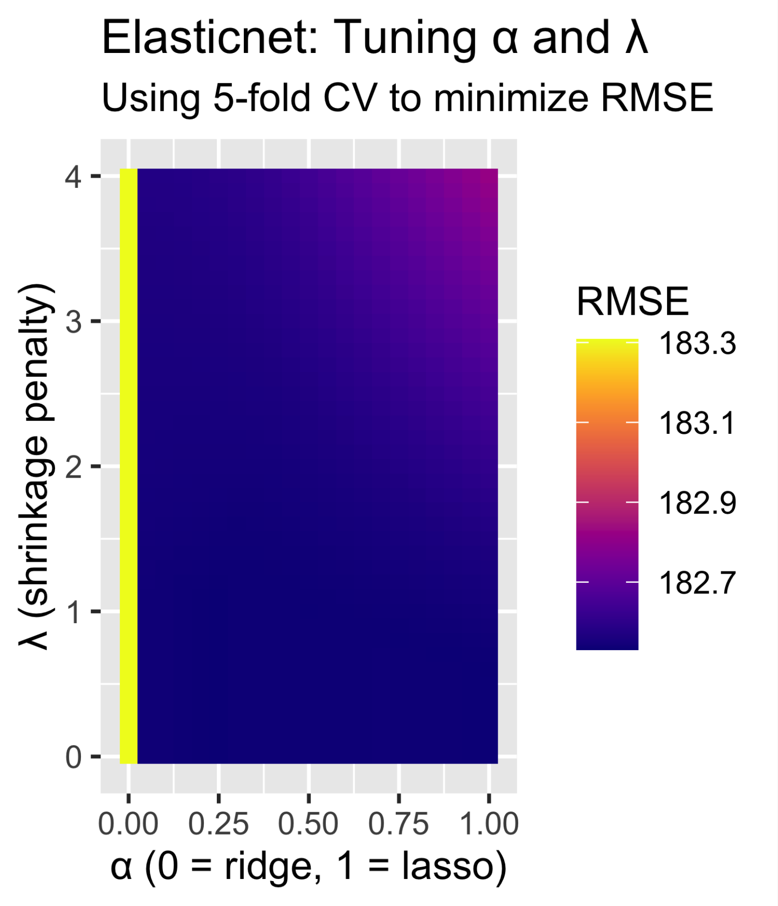
Expected performance:

- Not the best -- radiation is very non-linear with respect to time of day

Prediction Models

Training the elasticnet model

```r
# set a new seed for this chunk
set.seed(83352)

lambdas = seq(from = 0, to = 4, by = 0.1
alphas = seq(from = 0, to = 1, by = 0.05

elasticnet ← train(
  # the model: regress radiation on all
  radiation ~ .,
  data = train_std %>% select(-id),
  method = "glmnet",
  # evaluate performance with 5-fold crc
  trControl = trainControl("cv", number
  # the tuning parameters: alphas and la
  tuneGrid = expand.grid(
    alpha = alphas,
    lambda = lambdas
  )
)
```



Prediction Models

K-Nearst Neighbors model

How it works:

- Given an unlabeled observation of where we need to predict the radiation...
- Find the k closest labeled observations...
- The mean of their radiation values is our predicted radiation for the unlabeled observation

Parameters to tune:

- `k`: the number of neighbors to use

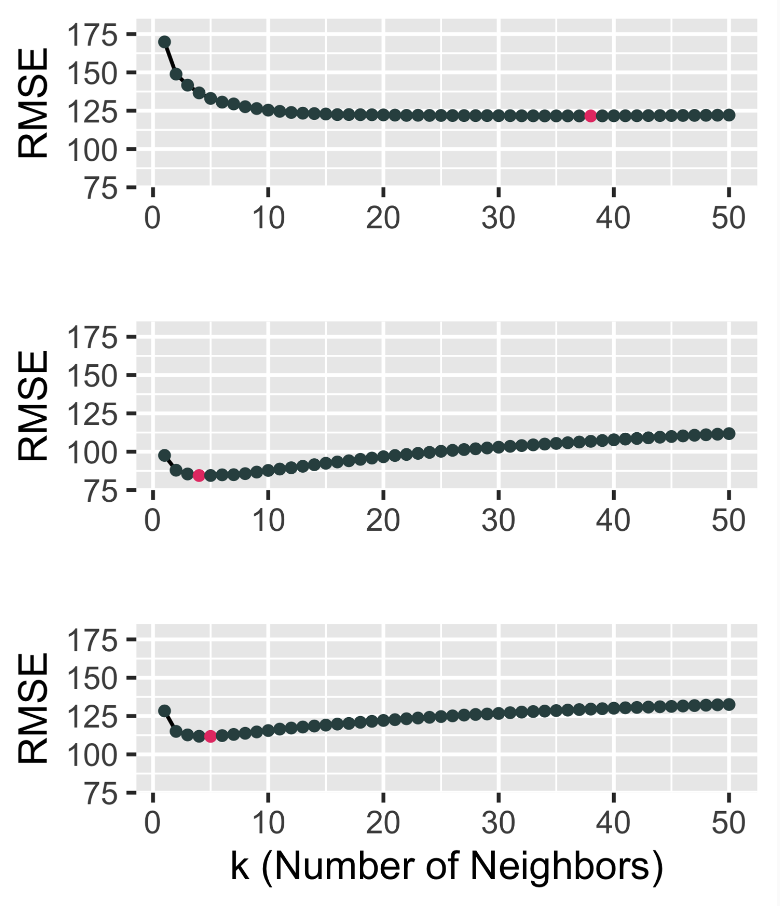
Expected performance:

- Better than any regression-based method -- does not require radiation to be linear with respect to predictors
- Beware of the curse of dimensionality

Prediction Models

Training three KNN models

```r
# set new seed for this code chunk
set.seed(86129)

knn_med ← train(
  # the model: predict radiation based o
  radiation ~ time + is_daytime + date +
    temperature + pressure + humidity,
  data = train_std %>% select(-id),
  method = "knn",
  # tune parameters using 5-fold cross-v
  trControl = trainControl("cv", number
  # tuning parameter: number of neighbor
  tuneGrid = expand.grid(k = seq(1, 50,
)
```



Tree-based models

How they work:

- Trees: at each step, find the best way to split the data (greedy algorithm)
- Forests: combine many individual trees
  - Create B bootstrapped samples
  - Train a tree on each sample, and at each split, only consider m variables
  - Aggregate across bootstrapped trees to get final model

Parameters to tune:

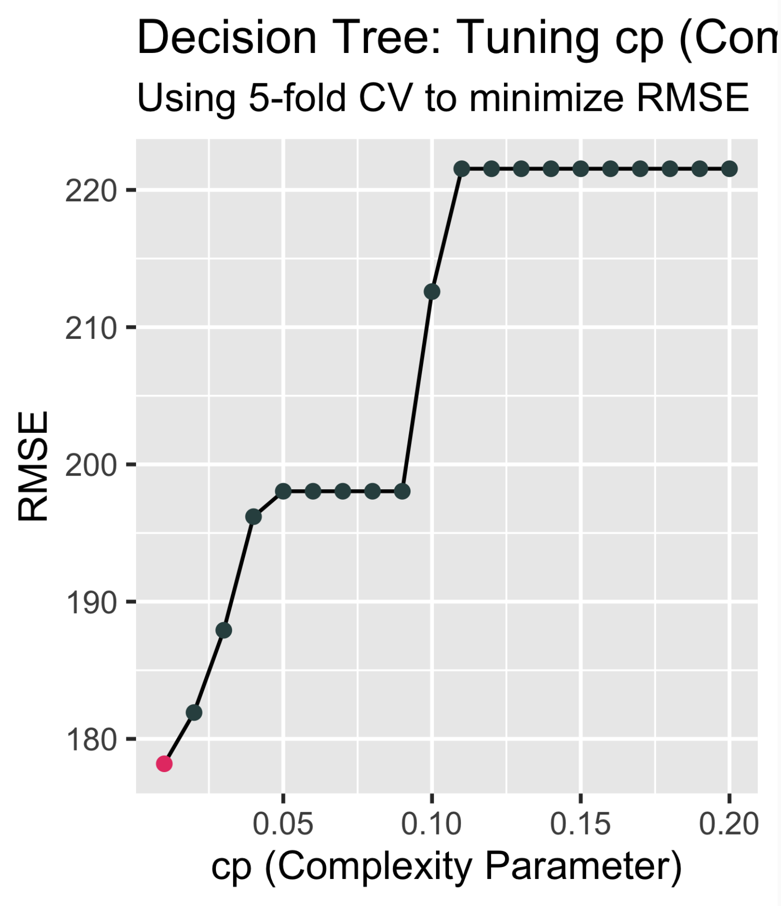
- `cp`: complexity parameter used for pruning
- `mtry`: number of variables to consider at each split
- `min.node.size`: the smallest number of observations allowed in a node

Expected performance:

- Single tree: probably better than elasticnet, not sure how it will compare to KNN
- Forest: better than any single tree, likely better than KNN

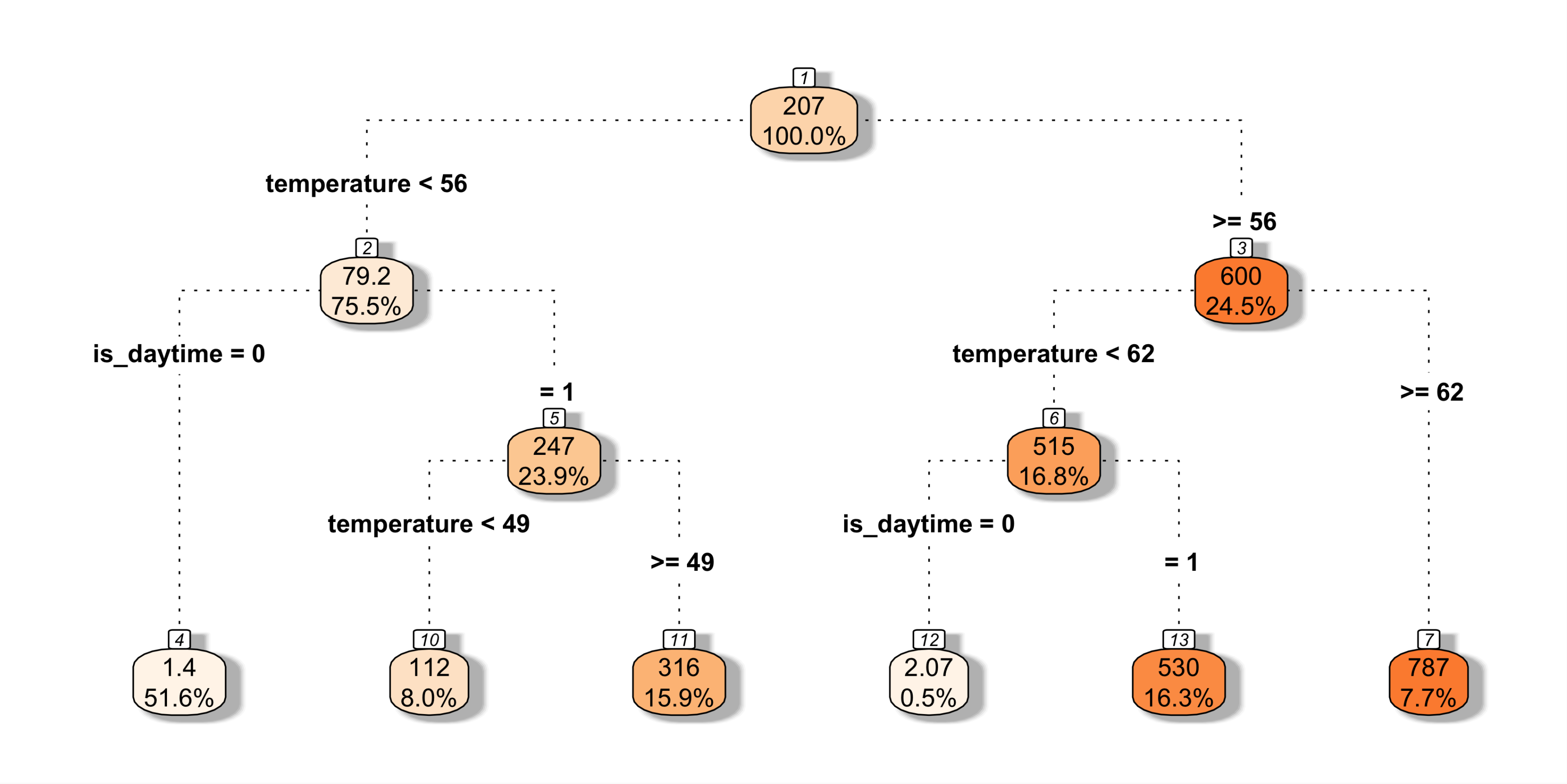
Training individual tree models

```
set.seed(64395)

tree_small <- train(
  # use only is_daytime and temp as prec
  radiation ~ .,
  data = train %>%
    select(is_daytime, temperature, radi

  method = "rpart",

  # tune cp using 5-fold cross-validatic
  trControl = trainControl("cv", number
  tuneGrid = data.frame(cp = seq(0.01, 0
)
```



Prediction Models

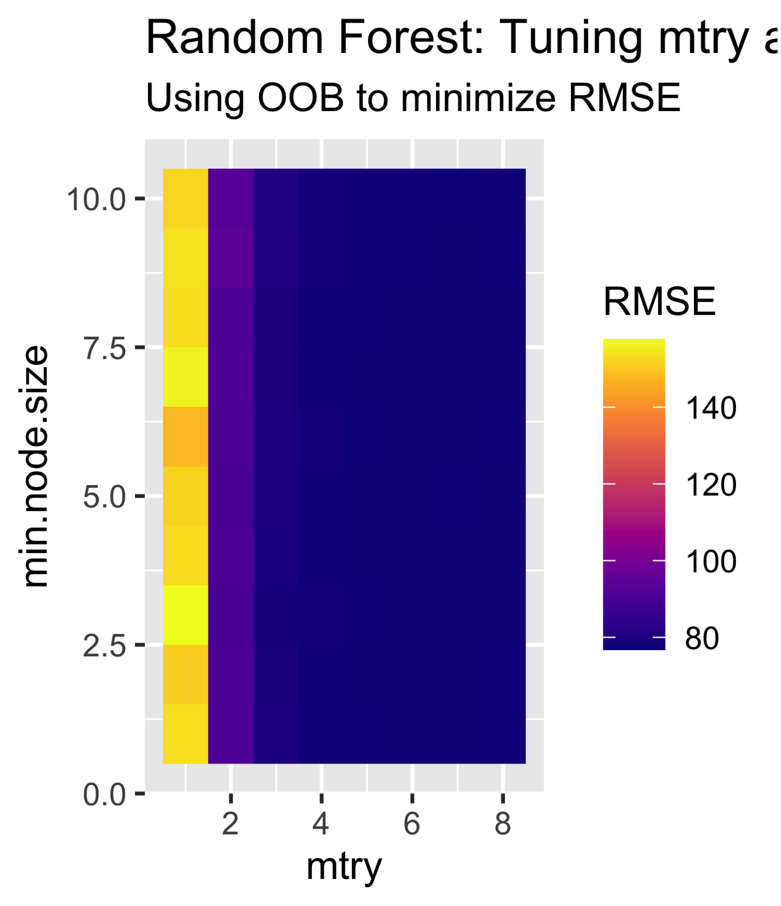
What does the tree look like?



Prediction Models

Training a random forest

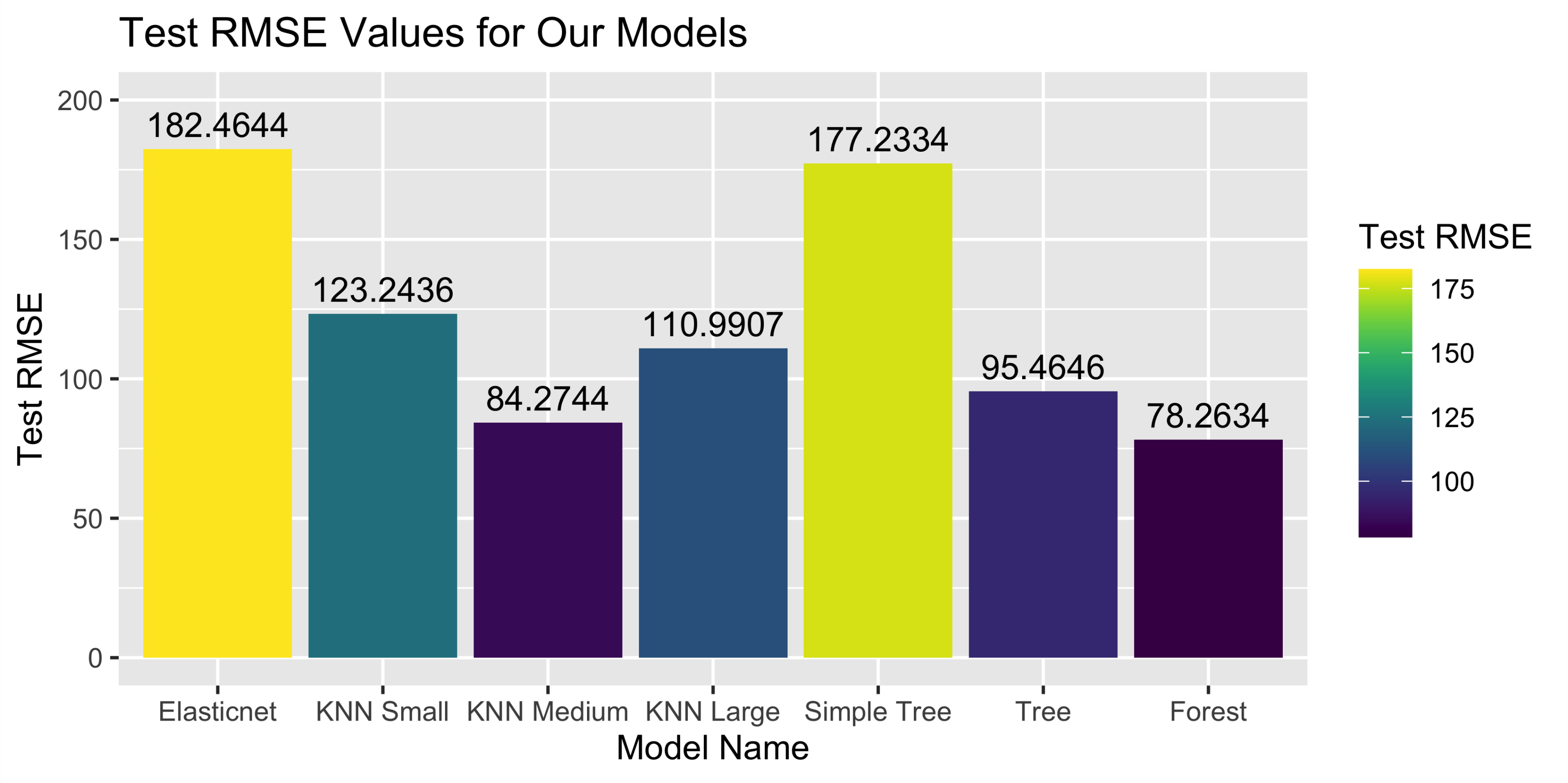
```
# set a new seed for this chunk
set.seed(42712)

# random forest model
forest = train(
  # The model: predict radiation based c
  radiation ~ .,
  # The data: non-standardized
  data = train %>% select(-id),
  # Implement random forest with 100 tre
  method = "ranger",
  num.trees = 100,
  # Evaluate performance with out-of-bag
  trControl = trainControl(method = "oob
  # Tuning parameters
  tuneGrid = expand.grid(
    "mtry" = c(1, 2, 3, 4, 5, 6, 7, 8),
    "splitrule" = "variance",
    "min.node.size" = 1:10
  )
)
```



Results

Results

How well did the models perform on data they have never seen?



Results

Conclusions

- As expected, the best model was the random forest and the worst was the elasticnet
- The standard deviation of radiation values in the data was 315.92 watts per square meter, and our best model had a test RMSE of 78.26 watts per square meter, about 1/4 of the standard deviation
- So our model predictions were pretty good, but not perfect