ML basics + classification

Divij 02-13

Classical ML

- Can be broken down into two main categories
  - Supervised ML
    - Regression - Predict **value** based on past
    - Classification - Predict **group** based on past
  - Unsupervised ML
    - Clustering - Break into **groups**
    - Dimensionality Reduction - “Important” components
    - Anomaly Detection - “Weirdness”

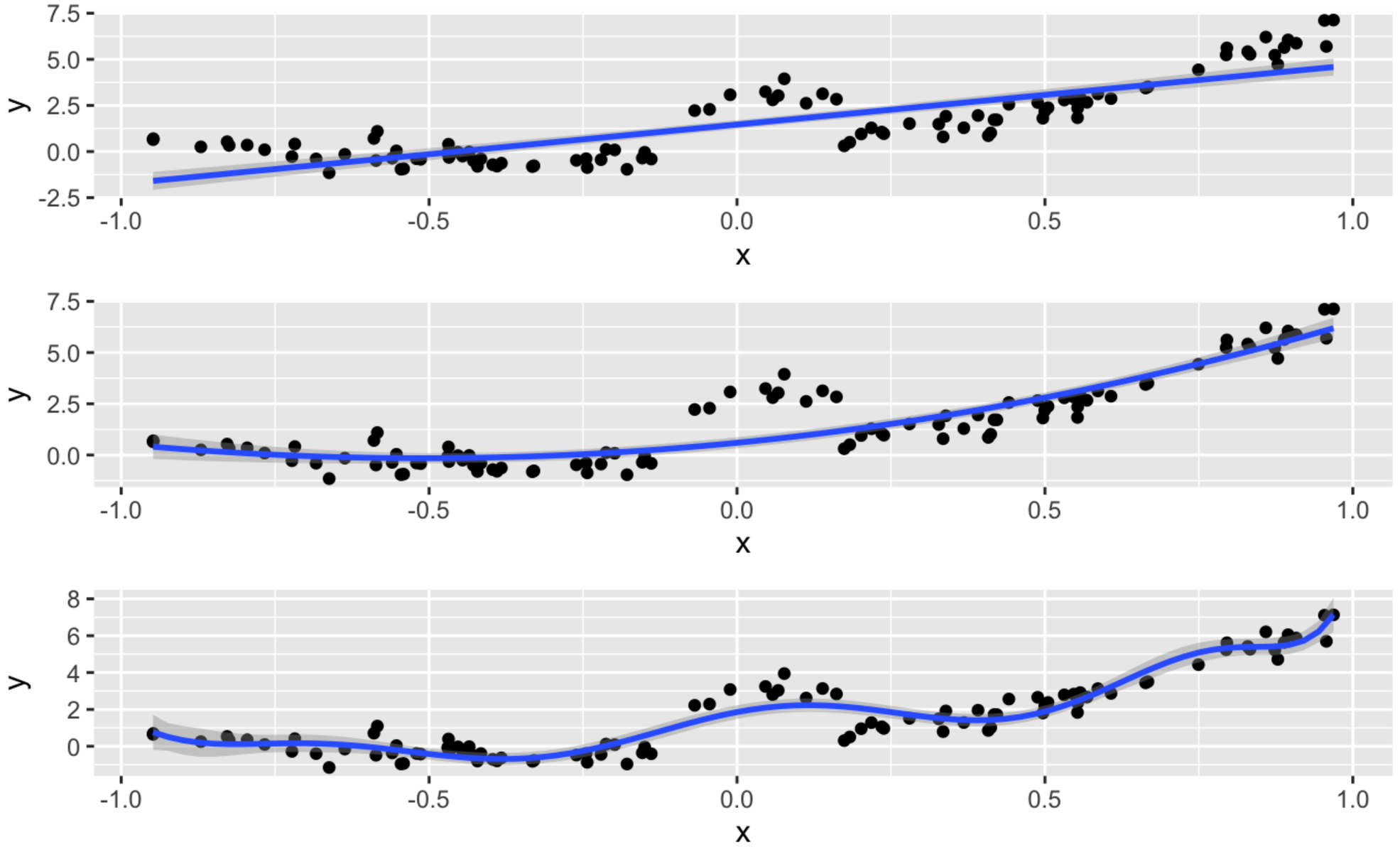
The process

- Let there be some ***feature(s)*** x
- Let there be some ***outcome*** y
- Then, there exist some ***hypothesis*** h of the relationship between x and y
- This hypothesis relies on ***parameters*** K
- Using some ***loss*** L that we try to minimize, we estimate the best ***parameters*** for the given ***hypothesis***

Parameters vs Hyperparameters

- **Parameters** - Parameters are the internal variables of a model that are learned from data during training.

- **Hyperparameters** - Hyperparameters are external settings chosen before training that control the learning process.



Underlying pattern vs sticking to points

Bias Variance tradeoff

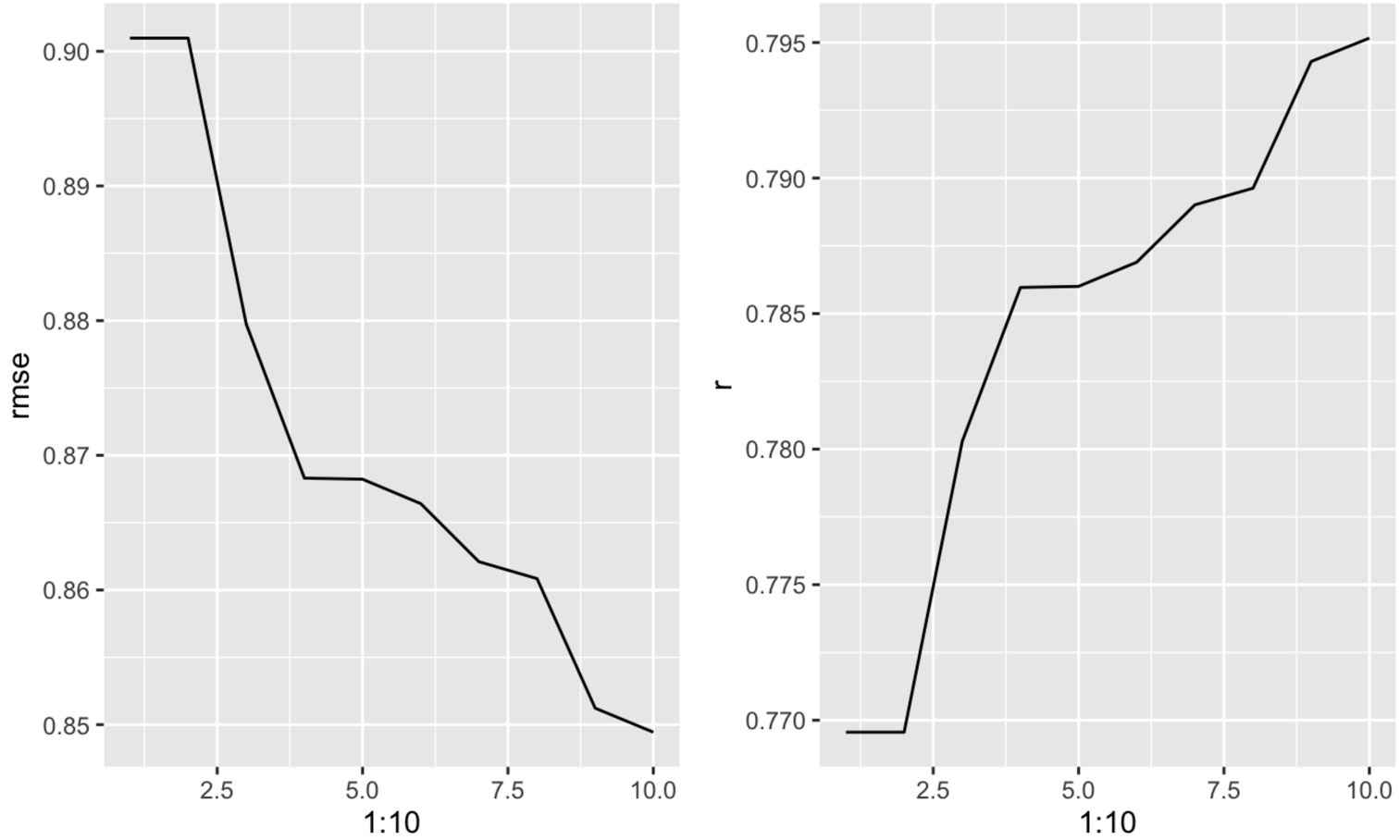
overfitting and underfitting

Ways to Combat

- Testing training split
- Cross validation split

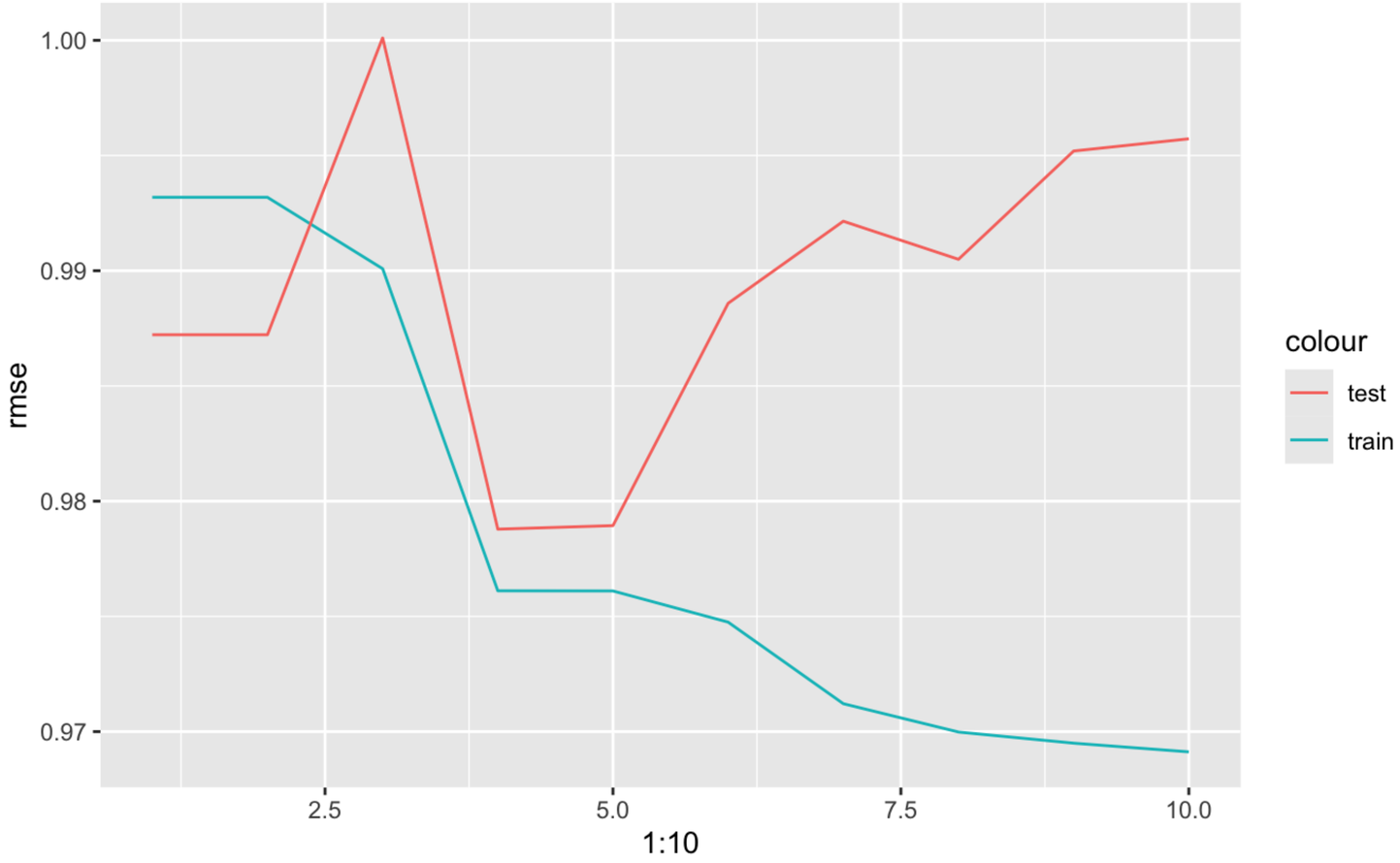
Training vs Test

Training on everything



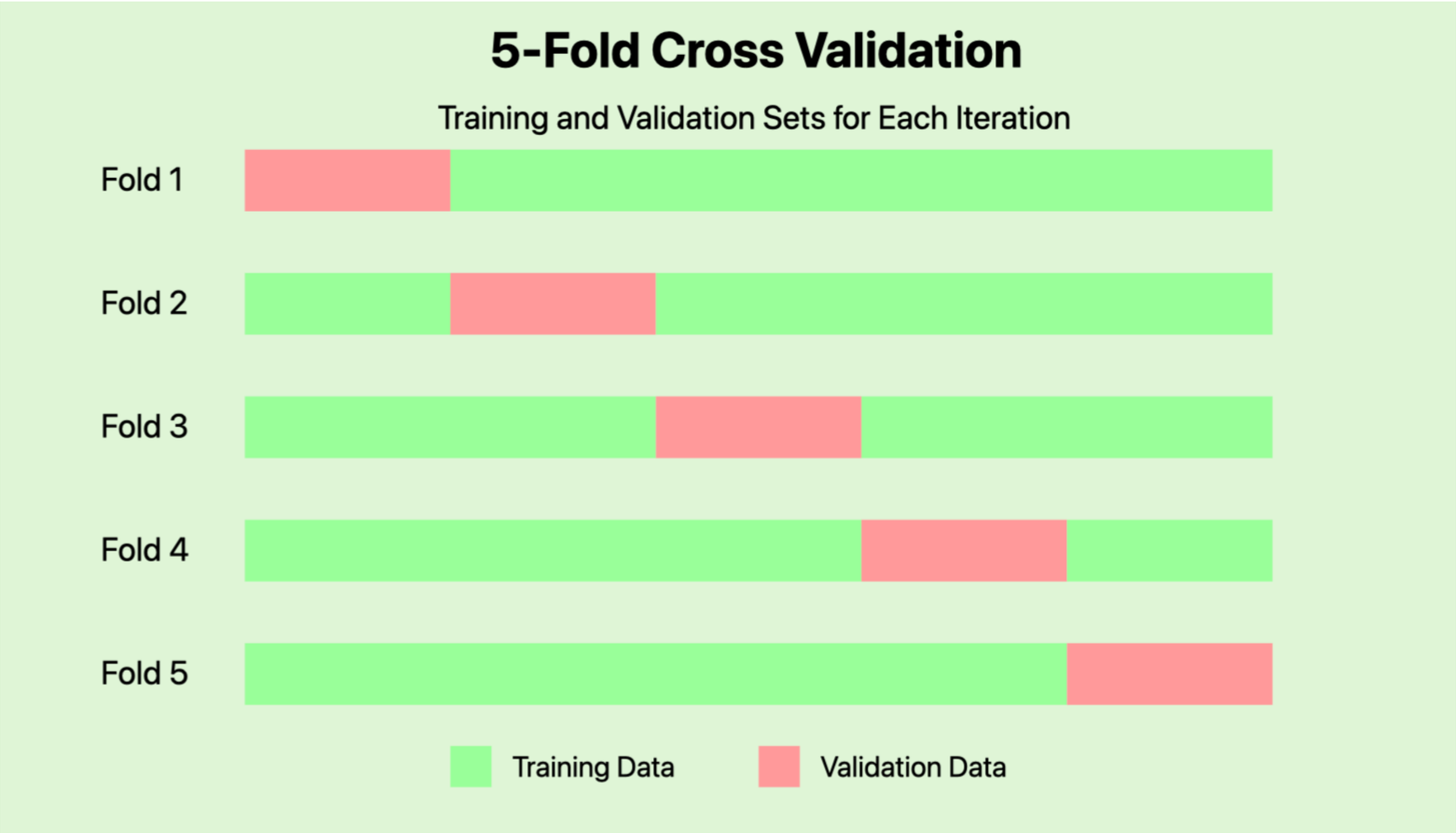
Training vs Test

Training vs testing 80-20



Cross Validation

Training on everything



Classification!

Logistic Regression

Health insurance coverage

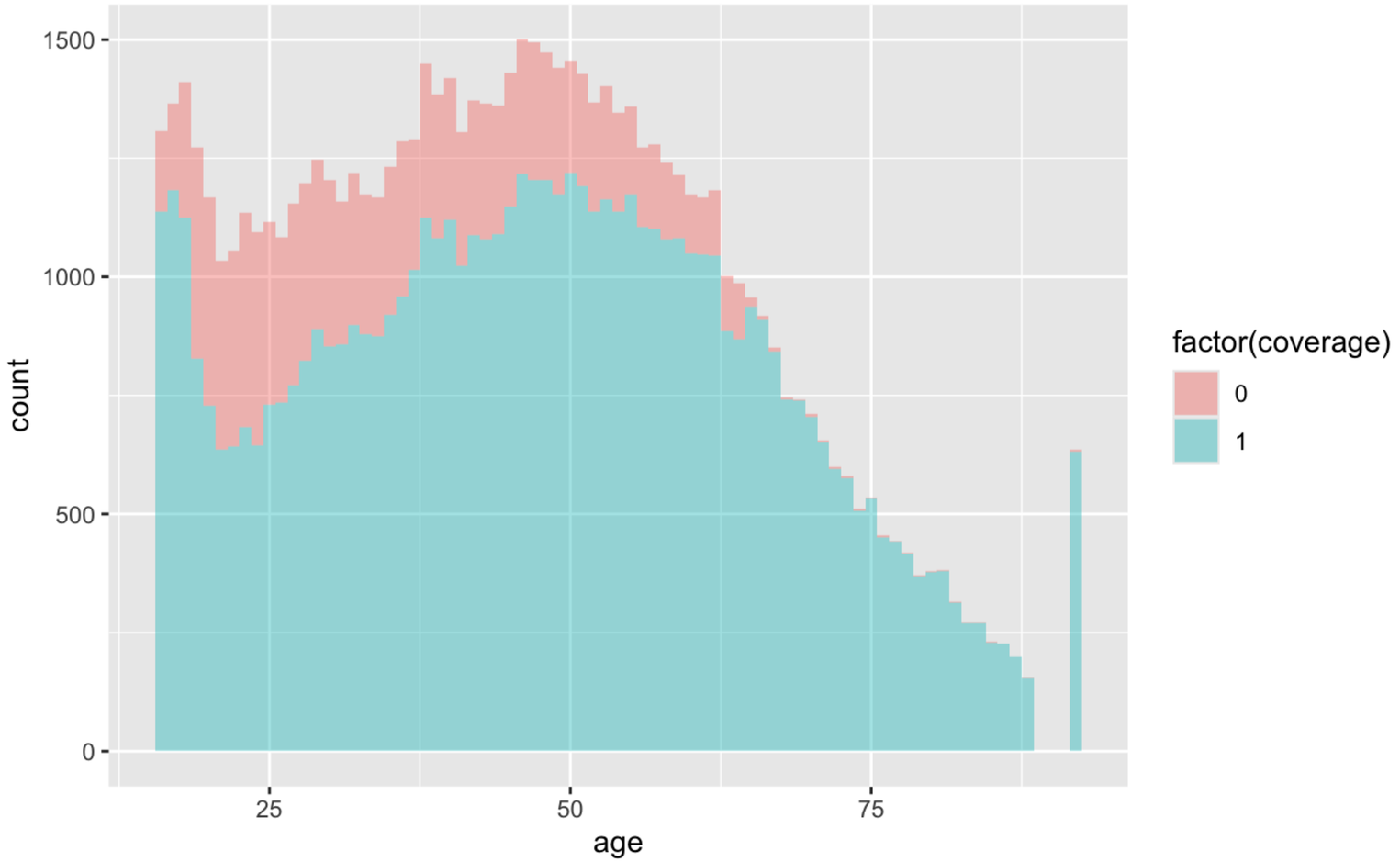
- Yes or No?
- What will we predict?

```
19
20 ```{r}
21 health %>%
22   pull(coverage) %>%
23   table()
24 ```
```

```
.
    0     1
13691 61114
```

Health insurance coverage

- Yes or No?
- Against Age



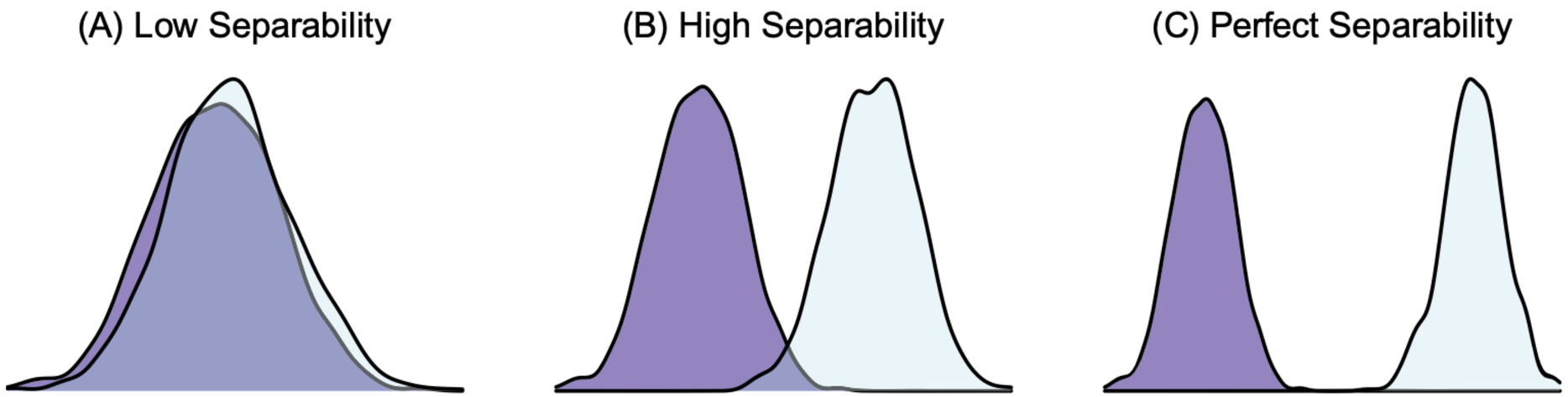
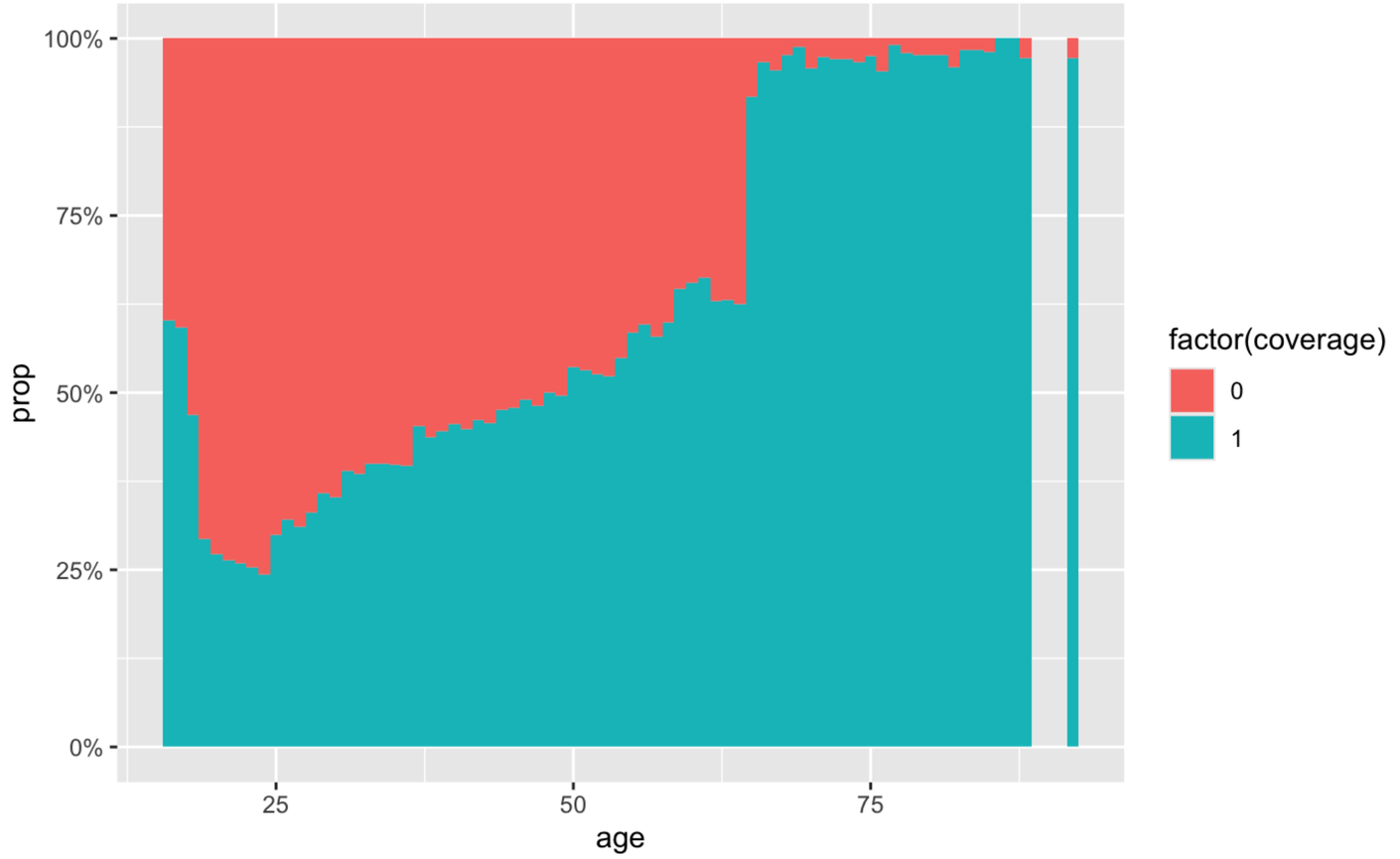


Figure 8.1: Separability can be easily seen in the distributions of a continuous variable when separated into two classes. These kernel density diagrams illustrate three scenarios: (A) Low Separability, (B) High Separability, (C) Perfect Separability.

Health insurance coverage

- Yes or No?
- Against Age



Health insurance coverage

- Yes or No?

```r
linear_reg() %>%
  fit(formula = coverage ~ age, data = health) %>%
  tidy()
```

A tibble: 2 × 5

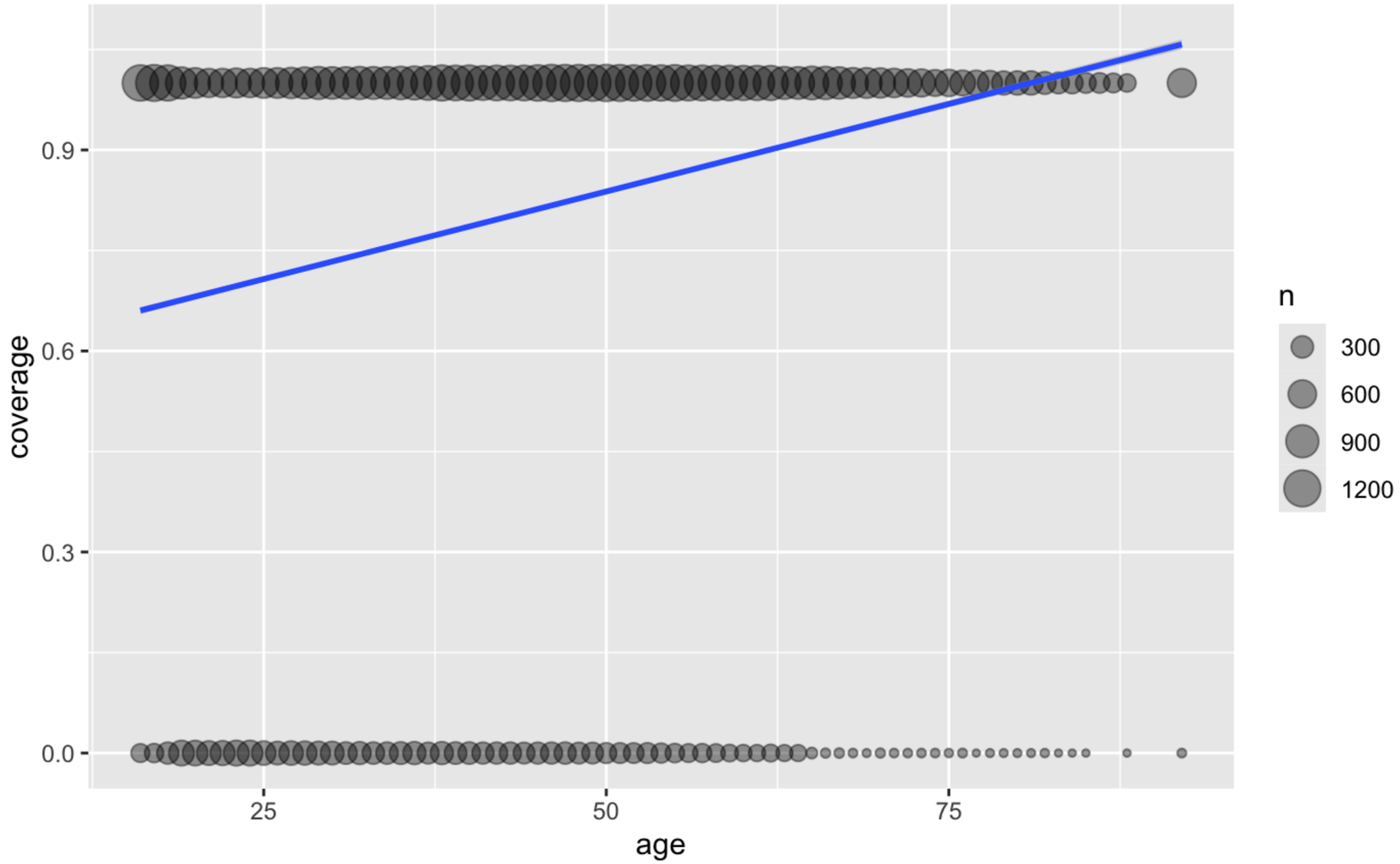
term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	0.576725845	3.729793e-03	154.62678	0
age	0.005225987	7.545483e-05	69.25981	0

2 rows

- Against Age

Health insurance coverage

- Yes or No?
- Against Age



Logistic Regression!

```r
log_reg_fit <- logistic_reg() %>%
  fit(formula = coverage ~ age, data = health)

log_reg_fit %>%
  tidy()
```

A tibble: 2 × 5

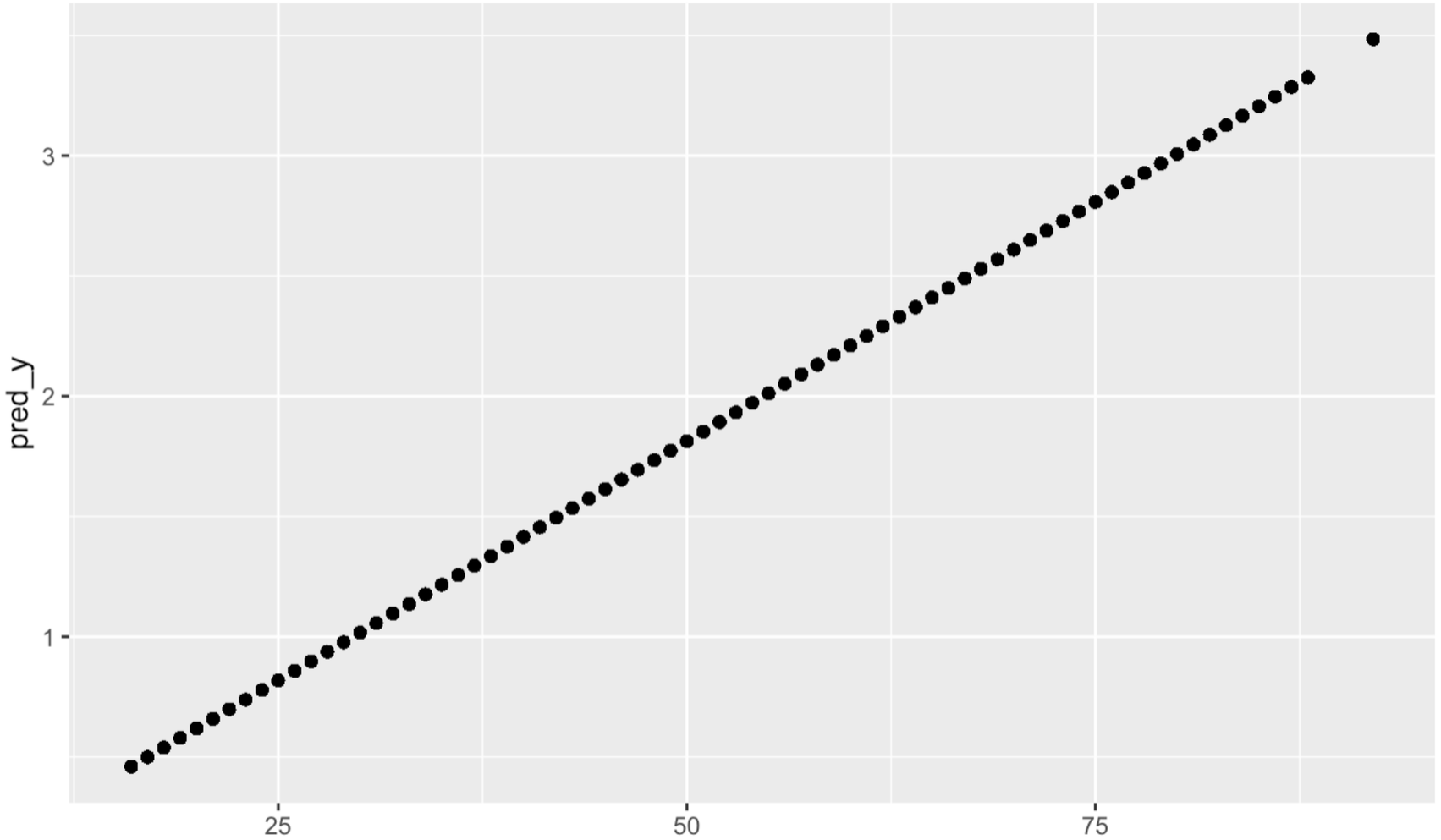
term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	-0.17773347	0.0255674024	-6.951565	3.612551e-12
age	0.03981371	0.0006140727	64.835498	0.000000e+00

2 rows

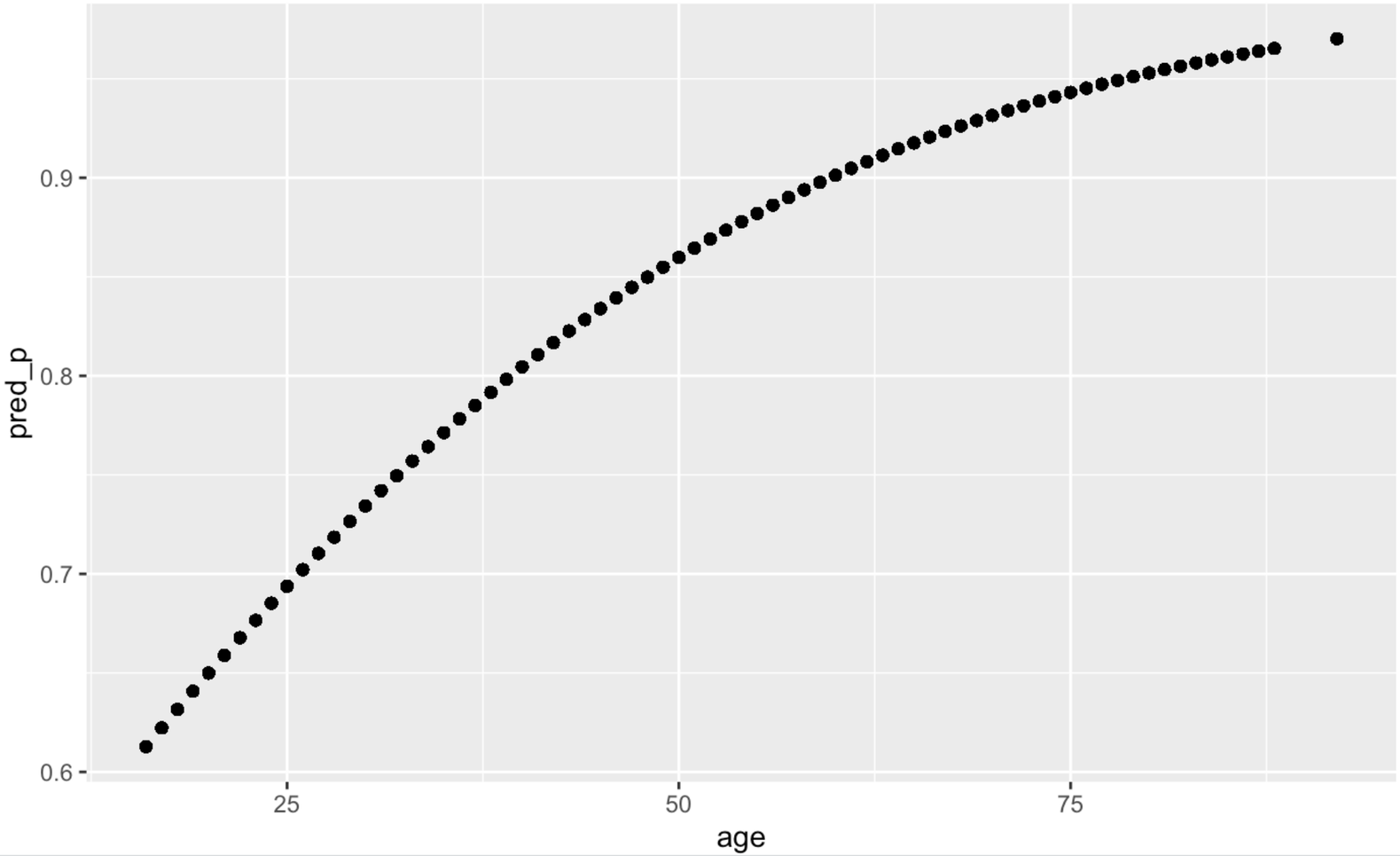
- Coefficient - “Average Change in Log Odds”
- e^coef - "Average Change in Odds”
- Probability of event -
  - Find y
  - p = e^y / (1 + e^y)

  - exp(x)

```{r}
health %>% mutate(pred_y = (age * 0.03981371) - 0.17773347) %>%
  ggplot(data = .) + geom_point(aes(x = age, y = pred_y))
```

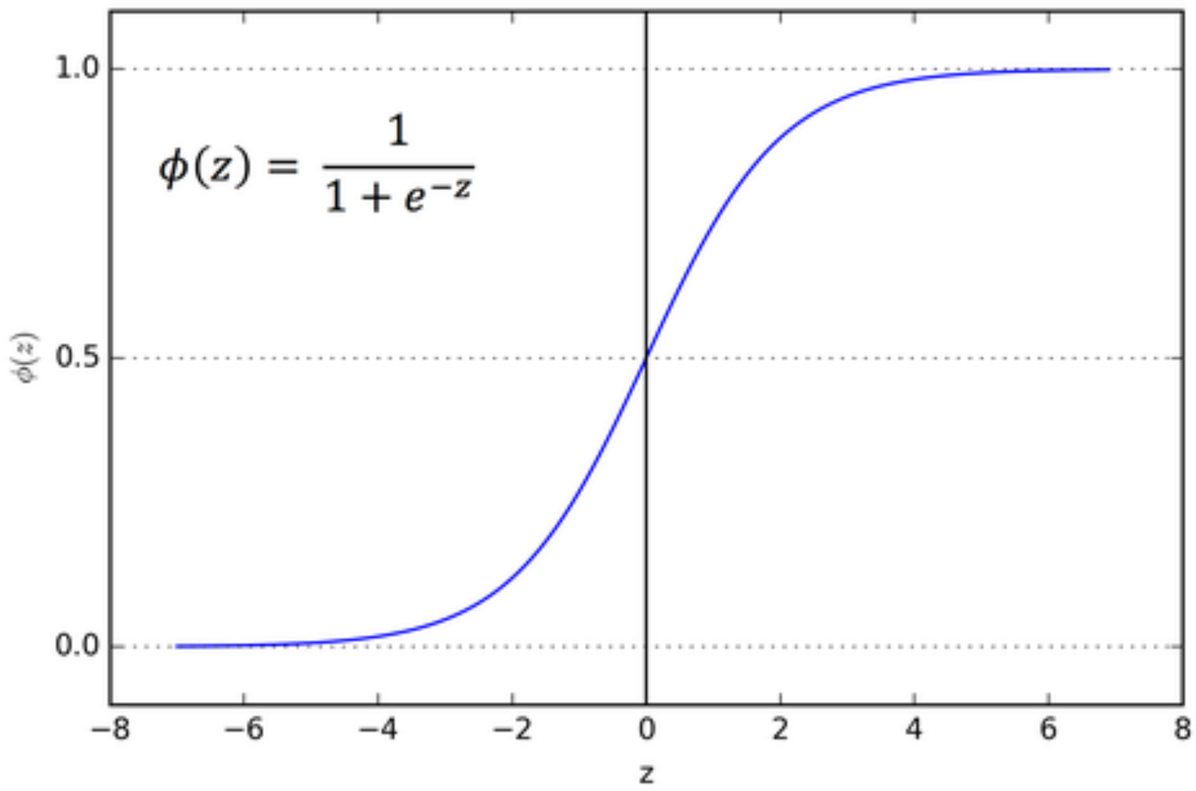


```r
health %>% mutate(pred_y = (age * 0.03981371) - 0.17773347) %>%
  mutate(pred_p = exp(pred_y) / (1 + exp(pred_y))) %>%
  ggplot(data = .) + geom_point(aes(x = age, y = pred_p))
```



What is this logistic?

- Sigmoid function - “S curve”
- Bounds the values between 0 and 1



87 ```

A tibble: 17 × 5

term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	1.17292366	0.2037067703	5.7579022	8.516568e-09
age	0.02987467	0.0008197871	36.4419831	9.215897e-291
raceAsian	0.22997884	0.1966444685	1.1695160	2.421958e-01
raceBlack	-0.07784148	0.1885716443	-0.4127952	6.797566e-01
raceNat. Hawaiian/Pac. Isl.	0.74787580	0.5266612659	1.4200319	1.555984e-01
raceOther	-0.72486863	0.1986325033	-3.6492951	2.629609e-04
raceTwo or More	0.26995172	0.2072153424	1.3027593	1.926569e-01
raceWhite	0.27370557	0.1881632136	1.4546179	1.457751e-01
sexMale	-0.30052090	0.0210534758	-14.2741703	3.170386e-46
citNon-citizen	-1.73990375	0.0416013990	-41.8232030	0.000000e+00

1-10 of 17 rows

Previous 1 2 Next

```
.
    0     1
3162 71643
```

Confusion Matrix

Table 8.3: Structure of confusion matrix.

	Predicted (F)	Predicted (T)
Actual (F)	True Negative (TN)	False Positive (FP)
Actual (T)	False Negative (FN)	True Positive (TP)

Table 8.4: A selection of classifier accuracy metrics derived from a confusion matrix.

Measure	Formula	Significance
True Positive Rate (TPR), Sensitivity, or Recall	$TPR = \frac{TP}{TP+FN}$	What proportion positive cases are correctly identified?
True Negative Rate (TNR) or Specificity	$TNR = \frac{TN}{TN+FP}$	What proportion negative cases are correctly identified?
False Positive Rate (FPR)	$FPR = \frac{FP}{FP+TN}$	What proportion of negative cases are incorrectly predicted as positive? Also known as Type I error rate.
False Negative Rate (FNR)	$FNR = \frac{FN}{TP+FN}$	Proportion of positive cases that are incorrectly predicted negative. Also known as the false alarm rate or Type II error rate.
Positive Predictive Value (PPV) or Precision	$PPV = \frac{TP}{TP+FP}$	The proportion of predicted positives will actually be positive? This measure is influenced by the prevalence of the outcome.

```{r}
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  table()
```

```
           coverage
.pred_class     0     1
          0  1926  1236
          1 11765 59878
```

```{r}
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  sensitivity(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
sensitivity	binary	0.9797755

1 row

```{r}
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  specificity(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
specificity	binary	0.1406764

1 row

```{r}
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  precision(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
precision	binary	0.835783

1 row

```{r}
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  recall(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
recall	binary	0.9797755

Measure	Formula	Significance
Accuracy (ACC)	$ACC = \frac{TP+TN}{n}$	The proportion of that records are correctly classified.
F1-Score (F1)	$F1 = \frac{2}{\frac{1}{TPR} \times \frac{1}{PPV}}$	Alternative method of calculating accuracy using a harmonic mean. It balances the TPR with the PPV.

```r
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  accuracy(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
accuracy	binary	0.8262015

1 row

```r
log_reg_fit %>% predict(health) %>%
  cbind(health) %>%
  select(.pred_class, coverage) %>%
  f_meas(truth = coverage, estimate = .pred_class, event_level = "second")
```

A tibble: 1 × 3

.metric <chr>	.estimator <chr>	.estimate <dbl>
f_meas	binary	0.9020692

1 row

About tidymodels

- Relies on 3 things
  - Workflows
  - Models
  - Recipes

Recipe

- For data preprocessing - getting it ready for a model

```
391  model_recipe <- recipe(sale.price ~ gross.square.feet + age + sale.year + borough,
392                         data = home_sales_nyc) %>%
393    step_center(all_numeric_predictors()) %>%
394    step_interact(terms = ~ gross.square.feet:age) %>%
395    step_poly(gross.square.feet, degree = 2) %>%
396    step_dummy(borough) %>%
397    step_normalize()
```

Models

- Choosing the model, and setting it up

```
178  model_spec <- rand_forest(mtry = 4, trees = 100) %>%
179    set_mode("classification")
180
```

Workflow

- Combination of models and recipes into one flow
  - Makes it easy to swap one or the other out
  - `workflow() %>%`
  - `  add_recipe(my_recipe) %>%`
  - `  add_model(my_model)`
  -

```{r}

# Split data
set.seed(123)
data_split <- initial_split(health, prop = 0.8)
train_data <- training(data_split)
test_data <- testing(data_split)

# Define logistic regression model
log_spec <- logistic_reg() %>%
  set_engine("glm")

# Create workflow
log_wf <- workflow() %>%
  add_model(log_spec) %>%
  add_formula(coverage ~ age + race + sex+ cit + mar + schl)

# Fit model
fit <- fit(log_wf, data = train_data)

# Evaluate model
preds <- predict(fit, test_data) %>%
  bind_cols(test_data)

sensitivity(preds, truth = coverage, estimate = .pred_class, event_level = "second")

```

What do we generally do in the ML process?

- Preprocess (clean, transform etc.) data, select features
- Choose a model
- Check quality of model fit based on error metrics, cross-validation, and generalization checks
- Iterate by tuning hyperparameters, adjusting features, and trying different models

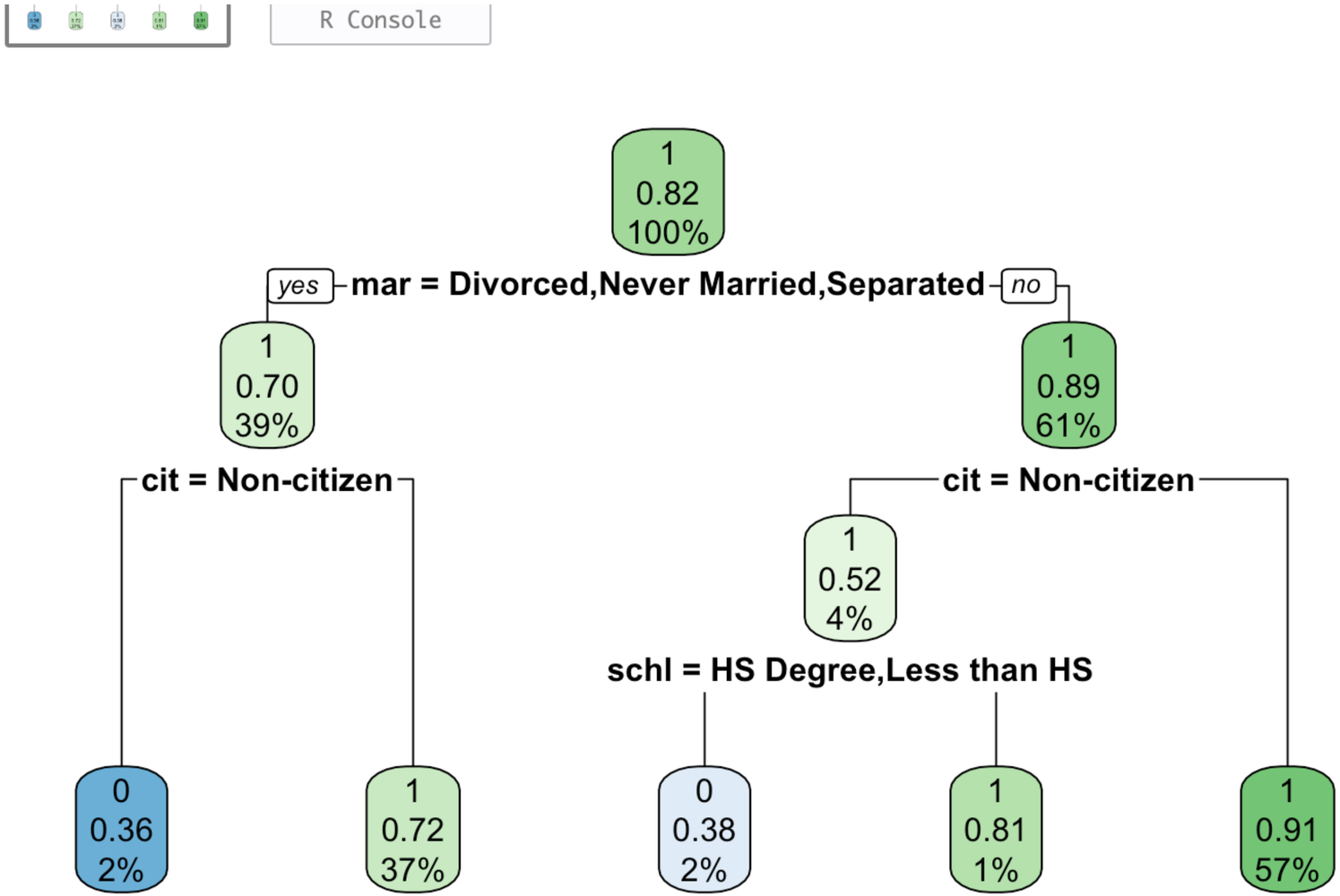
- Can be made easier by tuning hyper-parameters automatically
  - Next class

Two more models

- Random forest

- K nearest neighbors

Decision Tree



Random Forest

- A Random Forest is an ML model used for classification and regression tasks.
- It is an *ensemble* method that combines multiple decision trees to improve accuracy and reduce overfitting.
- Think of it as a "forest" where each tree makes a prediction, and the final decision is based on a majority vote (classification) or an average prediction (regression).

- For classification: Each tree votes on a class, and the most common class is chosen.
- For regression: The average prediction across all trees is taken as the final result.

K-nearest neighbors

- KNN is a lazy learning algorithm, meaning it does not learn a model in advance but makes predictions by comparing new data points to stored examples.

- Find the K nearest data points to the new input based on a distance metric (e.g., Euclidean distance).
- For classification: Assign the most common class among the neighbors.
- For regression: Take the average value of the neighbors.

What is distance?