Include TSNE Output

Simulate some data

The distribution of the response depends on both a latent variable t (which is recovered fairly accurately by T-SNE) and one of the observed covariates $x1$.

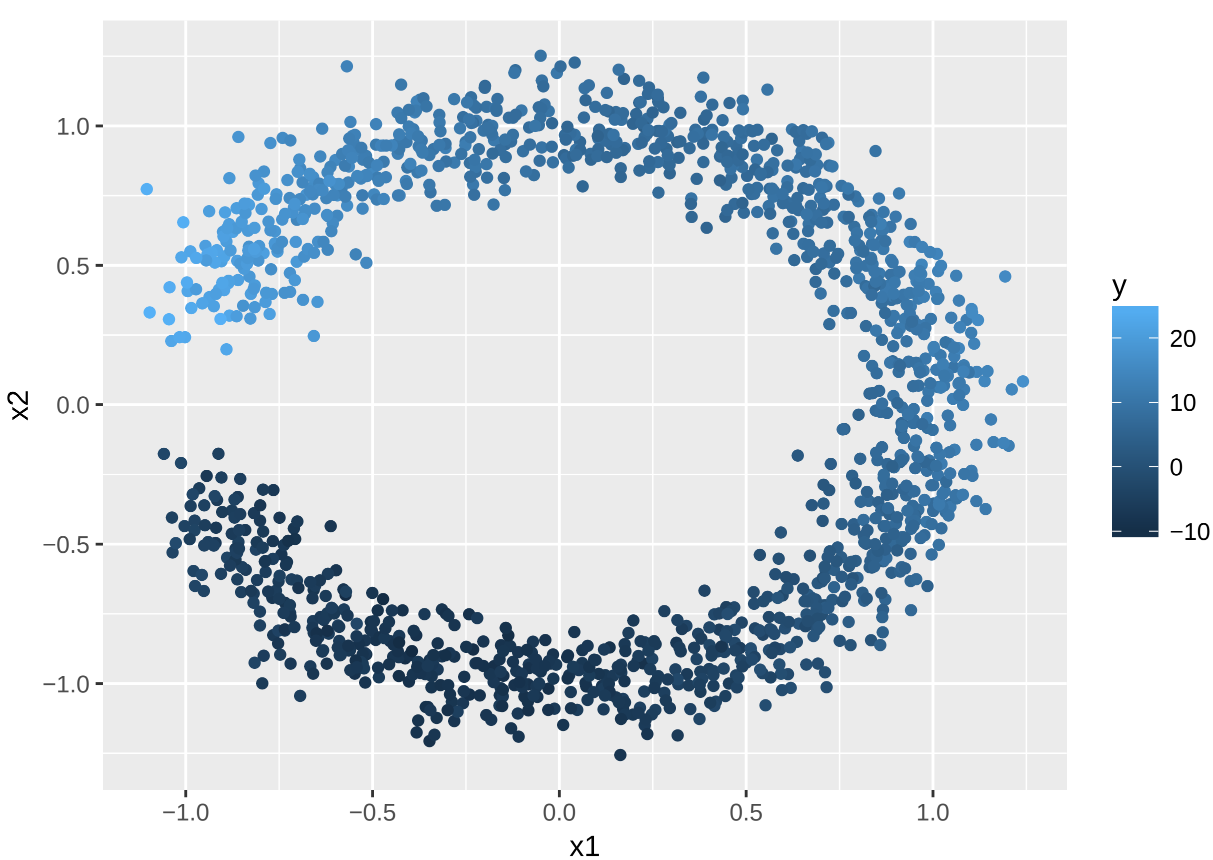
```
library(Rtsne)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(purrr)
library(caret)
```

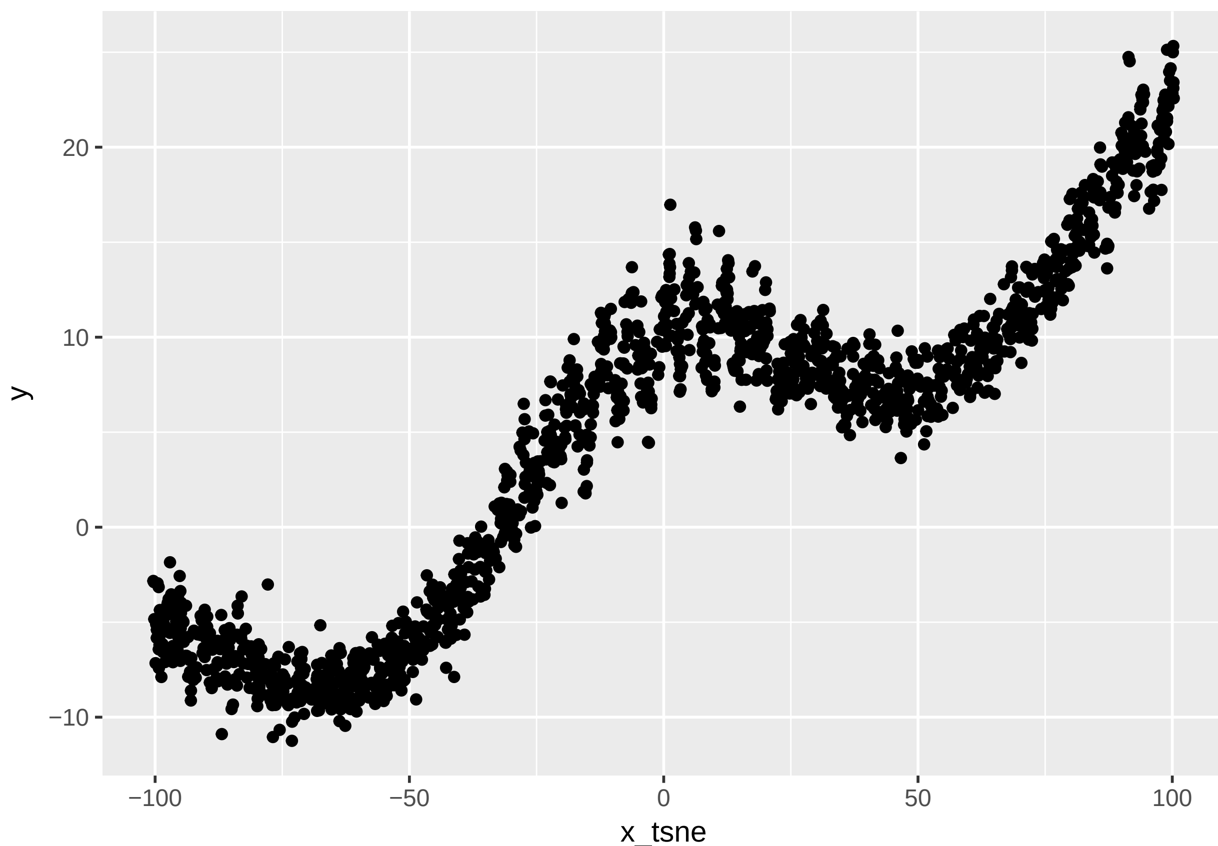
```
## Loading required package: lattice

## Loading required package: ggplot2

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(47927)

n <- 1500
t_limit <- 2.8
t <- runif(n = n, -t_limit, t_limit)

example_data <- data.frame(
    x1 = cos(t) + rnorm(n, 0, 0.1),
    x2 = sin(t) + rnorm(n, 0, 0.1)
  ) %>%
  mutate(
    y = 5 * t + 10 * x1^2 + rnorm(n, 0, 1)
  )

ggplot(data = example_data,
    mapping = aes(x = x1, y = x2, color = y)) +
  geom_point()
```

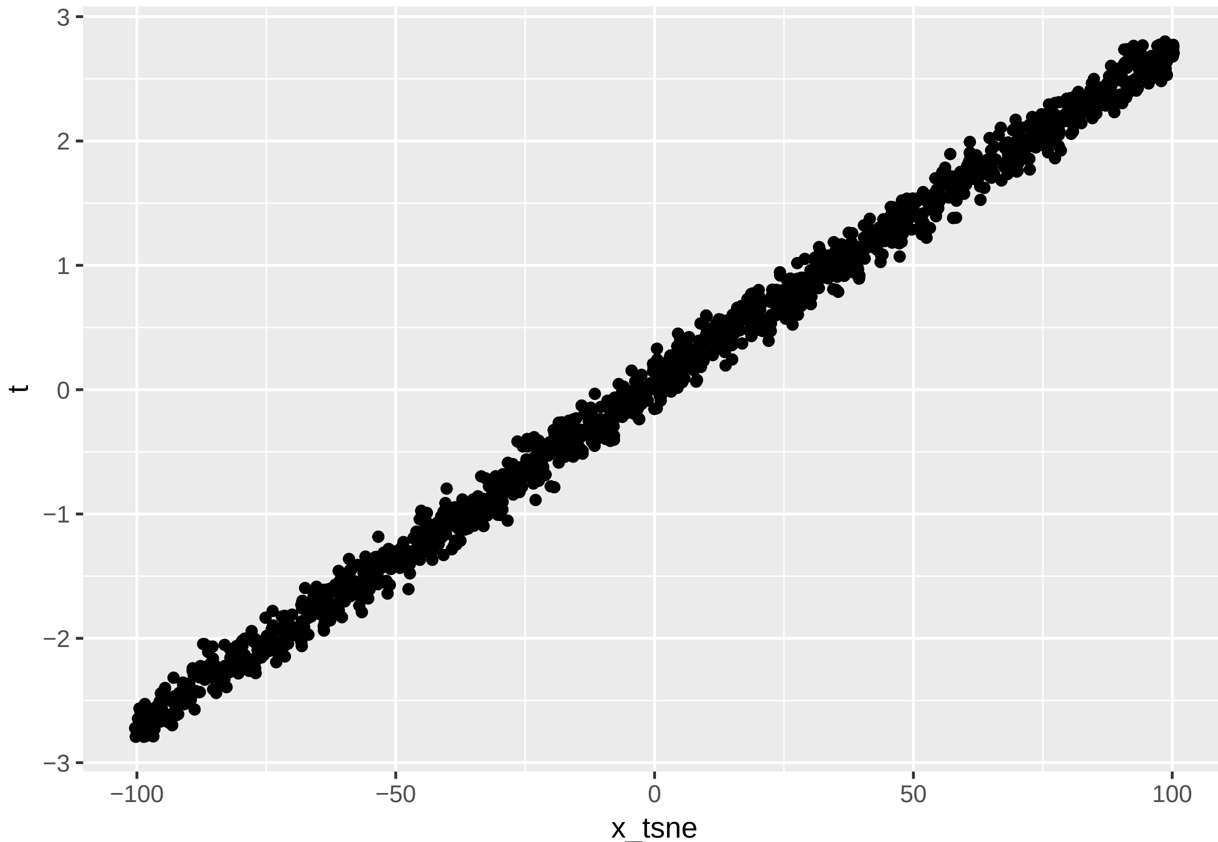


TSNE to reduce to 1 dimension

```
tsne_result = Rtsne(
  example_data %>%
    dplyr::select(x1, x2) %>%
    as.matrix(),
  pca=FALSE,
  theta=0,
  dims=1)

example_data <- example_data %>%
  mutate(
    x_tsne = tsne_result$Y[, 1]
  )

# plot showing association between x_tsne and response
ggplot(data = example_data, mapping = aes(x = x_tsne, y = y)) +
  geom_point()
```




```r
# plot showing association between x_tsne and t
ggplot(data = example_data, mapping = aes(x = x_tsne, y = t)) +
  geom_point()
```



Compare performance of 3 approaches

Train/test and cross-validation splits

```
# train/test split
tt_inds <- caret::createDataPartition(
  example_data$y,
  p = 0.5
)

example_train <- example_data %>%
  slice(tt_inds[[1]])

example_test <- example_data %>%
  slice(-tt_inds[[1]])

## 10-fold cross-validation splits
crossval_val_fold_inds <- caret::createFolds(example_train$y, k = 10)

get_complementary_inds <- function(val_inds) {
  return(seq_len(nrow(example_train))[-val_inds])
}

crossval_train_fold_inds <- purrr::map(
  crossval_val_fold_inds,
```

```
  get_complementary_inds
)
```

Gradient Tree Boosting fit based on original variables

```
set.seed(98364)

# gtb based on original explanatory variables (x1, x2)
xgb_fit_original <- train(
  y ~ x1 + x2,
  data = example_train,
  method = "xgbTree",
  trControl = trainControl(
    method = "cv",
    number = 10,
    index = crossval_train_fold_inds,
    indexOut = crossval_val_fold_inds),
  tuneGrid = expand.grid(
    nrounds = c(5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200),
    eta = c(0.01, 0.05, 0.1), # learning rate; 0.3 is the default
    gamma = 0, # minimum loss reduction to make a split; 0 is the default
    max_depth = 1:5, # how deep are our trees?
    subsample = c(0.4, 0.5, 0.9, 1), # proportion of observations to use in growing each tree
    colsample_bytree = 1, # proportion of explanatory variables used in each tree
    min_child_weight = 1 # think of this as how many observations must be in each leaf node
  )
)

xgb_fit_original$results %>%
  filter(RMSE == min(RMSE))
```

```
##   eta max_depth gamma colsample_bytree min_child_weight subsample nrounds
## 1 0.1         4     0                1                1       0.9      70
##       RMSE  Rsquared       MAE   RMSESD  RsquaredSD     MAESD
## 1 1.201088 0.9818886 0.9554054 0.108413 0.002612028 0.1015137
```

```
mean((example_test$y - predict(xgb_fit_original, example_test))^2)
```

```
## [1] 1.580147
```

```
var_importance_original <- varImp(xgb_fit_original, scale = FALSE)
var_importance_original$importance
```

```
##      Overall
## x2 0.8023147
## x1 0.1976853
```

Gradient tree boosting based on original explanatory variables and outputs from TSNE

```
# gtb based on original explanatory variables (x1, x2)
set.seed(98364)

xgb_fit_original_and_tsne <- train(
```

```r
  y ~ x1 + x2 + x_tsne,
  data = example_train,
  method = "xgbTree",
  trControl = trainControl(
    method = "cv",
    number = 10,
    index = crossval_train_fold_inds,
    indexOut = crossval_val_fold_inds),
  tuneGrid = expand.grid(
    nrounds = c(5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200, 250, 300, 350, 400, 450
    eta = c(0.01, 0.05, 0.1), # learning rate; 0.3 is the default
    gamma = 0, # minimum loss reduction to make a split; 0 is the default
    max_depth = 1:5, # how deep are our trees?
    subsample = c(0.4, 0.5, 0.9, 1), # proportion of observations to use in growing each tree
    colsample_bytree = 1, # proportion of explanatory variables used in each tree
    min_child_weight = 1 # think of this as how many observations must be in each leaf node
  )
)

xgb_fit_original_and_tsne$results %>%
  filter(RMSE == min(RMSE))
```

```
##    eta max_depth gamma colsample_bytree min_child_weight subsample nrounds
## 1 0.05         4     0                1                1       0.5     175
##       RMSE  Rsquared       MAE     RMSESD  RsquaredSD      MAESD
## 1 1.183495 0.9824602 0.9437638 0.07521843 0.001687728 0.06802422
```

```r
mean((example_test$y - predict(xgb_fit_original_and_tsne, example_test))^2)
```

```
## [1] 1.452459
```

```r
var_importance_original_and_tsne <- varImp(xgb_fit_original_and_tsne, scale = FALSE)
var_importance_original_and_tsne$importance
```

```
##           Overall
## x_tsne 0.84987417
## x1     0.13995338
## x2     0.01017245
```

Gradient tree boosting based on outputs from TSNE only

```r
# gtb based on original explanatory variables (x1, x2)
set.seed(98364)

xgb_fit_tsne <- train(
  y ~ x_tsne,
  data = example_train,
  method = "xgbTree",
  trControl = trainControl(
    method = "cv",
    number = 10,
    index = crossval_train_fold_inds,
    indexOut = crossval_val_fold_inds),
  tuneGrid = expand.grid(
```

```
    nrounds = c(5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200, 250, 300, 350, 400),
    eta = c(0.01, 0.05, 0.1, 0.2), # learning rate; 0.3 is the default
    gamma = 0, # minimum loss reduction to make a split; 0 is the default
    max_depth = 1:5, # how deep are our trees?
    subsample = c(0.2, 0.3, 0.4, 0.5), # proportion of observations to use in growing each tree
    colsample_bytree = 1, # proportion of explanatory variables used in each tree
    min_child_weight = 1 # think of this as how many observations must be in each leaf node
  )
)

xgb_fit_tsne$results %>%
  filter(RMSE == min(RMSE))
```

```
##   eta max_depth gamma colsample_bytree min_child_weight subsample nrounds
## 1 0.1         3     0                1                1       0.5     200
##       RMSE  Rsquared      MAE     RMSESD  RsquaredSD      MAESD
## 1 1.335491 0.9777101 1.064715 0.08735691 0.002452403 0.06415724
```

```
mean((example_test$y - predict(xgb_fit_tsne, example_test))^2)
```

```
## [1] 1.871567
```

```
var_importance_tsne <- varImp(xgb_fit_tsne, scale = FALSE)
var_importance_tsne$importance
```

```
##        Overall
## x_tsne       1
```

Differences between each pair of models are statistically significant

```
orig_squared_errors <- (example_test$y - predict(xgb_fit_original, example_test))^2
orig_and_tsne_squared_errors <- (example_test$y - predict(xgb_fit_original_and_tsne, example_test))^2
tsne_squared_errors <- (example_test$y - predict(xgb_fit_tsne, example_test))^2
```

```
t.test(orig_squared_errors, orig_and_tsne_squared_errors, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  orig_squared_errors and orig_and_tsne_squared_errors
## t = 3.8191, df = 747, p-value = 0.000145
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.06205255 0.19332378
## sample estimates:
## mean of the differences
##               0.1276882
```

```
t.test(orig_squared_errors, tsne_squared_errors, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  orig_squared_errors and tsne_squared_errors
## t = -4.2021, df = 747, p-value = 2.964e-05
```

```
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.4275641 -0.1552742
## sample estimates:
## mean of the differences
##              -0.2914192
```

```r
t.test(orig_and_tsne_squared_errors, tsne_squared_errors, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  orig_and_tsne_squared_errors and tsne_squared_errors
## t = -6.6101, df = 747, p-value = 7.307e-11
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.5435786 -0.2946361
## sample estimates:
## mean of the differences
##              -0.4191073
```