Google_Insurance

Ana

2023-05-07

USE

ACME Insurance Inc. offers affordable health insurance to thousands of customer all over the United States. You're tasked with creating an automated system to estimate the annual medical expenditure for new customers, using information such as their age, sex, BMI, children, smoking habits and region of residence.

```
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
library(ggplot2)
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.1     v tidyr     1.3.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

Exploring data

```
library(readr)
#dataset <- read_csv('00-insurance.csv')
dataset <- read_csv('00-insurance.csv', col_types = cols(
  sex = col_character(),
  smoker = col_character(),
  region = col_character(),
  age = col_double(),
  bmi = col_double(),
  children = col_double(),
  charges = col_double()
))
```

```
head(dataset)
```

```
## # A tibble: 6 x 7
##     age sex      bmi children smoker region    charges
##   <dbl> <chr>  <dbl>    <dbl> <chr>  <chr>       <dbl>
## 1    19 female  27.9        0 yes    southwest  16885.
## 2    18 male    33.8        1 no     southeast   1726.
## 3    28 male    33          3 no     southeast   4449.
## 4    33 male    22.7        0 no     northwest  21984.
## 5    32 male    28.9        0 no     northwest   3867.
## 6    31 female  25.7        0 no     southeast   3757.
```

```
str(dataset)
```

```
## spc_tbl_ [1,338 x 7] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ age     : num [1:1338] 19 18 28 33 32 31 46 37 37 60 ...
##  $ sex     : chr [1:1338] "female" "male" "male" "male" ...
##  $ bmi     : num [1:1338] 27.9 33.8 33 22.7 28.9 ...
##  $ children: num [1:1338] 0 1 3 0 0 0 1 3 2 0 ...
##  $ smoker  : chr [1:1338] "yes" "no" "no" "no" ...
##  $ region  : chr [1:1338] "southwest" "southeast" "southeast" "northwest" ...
##  $ charges : num [1:1338] 16885 1726 4449 21984 3867 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   age = col_double(),
##   ..   sex = col_character(),
##   ..   bmi = col_double(),
##   ..   children = col_double(),
##   ..   smoker = col_character(),
##   ..   region = col_character(),
##   ..   charges = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
glimpse(dataset)
```

```
## Rows: 1,338
## Columns: 7
## $ age      <dbl> 19, 18, 28, 33, 32, 31, 46, 37, 37, 60, 25, 62, 23, 56, 27, 1~
## $ sex      <chr> "female", "male", "male", "male", "male", "female", "female",~
## $ bmi      <dbl> 27.900, 33.770, 33.000, 22.705, 28.880, 25.740, 33.440, 27.74~
## $ children <dbl> 0, 1, 3, 0, 0, 0, 1, 3, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0~
## $ smoker   <chr> "yes", "no", "no", "no", "no", "no", "no", "no", "no", "no", ~
## $ region   <chr> "southwest", "southeast", "southeast", "northwest", "northwes~
## $ charges  <dbl> 16884.924, 1725.552, 4449.462, 21984.471, 3866.855, 3756.622,~
```

```
nulo<-is.na(dataset)
sum(nulo)
```

```
## [1] 0
```

```
summary(dataset)
```

```
##       age            sex                 bmi           children    
##  Min.   :18.00   Length:1338        Min.   :15.96   Min.   :0.000  
##  1st Qu.:27.00   Class :character   1st Qu.:26.30   1st Qu.:0.000  
##  Median :39.00   Mode  :character   Median :30.40   Median :1.000  
##  Mean   :39.21                      Mean   :30.66   Mean   :1.095  
##  3rd Qu.:51.00                      3rd Qu.:34.69   3rd Qu.:2.000  
##  Max.   :64.00                      Max.   :53.13   Max.   :5.000
##     smoker             region             charges
##  Length:1338        Length:1338        Min.   : 1122
##  Class :character   Class :character   1st Qu.: 4740
##  Mode  :character   Mode  :character   Median : 9382
##                                        Mean   :13270
##                                        3rd Qu.:16640
##                                        Max.   :63770
```

```
duplicates<-duplicated(dataset)
sum(duplicates)#number of duplicates
```

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

```
filter(dataset,duplicates)
```

```
## # A tibble: 1 x 7
##     age sex     bmi children smoker region    charges
##   <dbl> <chr> <dbl>    <dbl> <chr>  <chr>       <dbl>
## 1    19 male   30.6        0 no     northwest   1640.
```

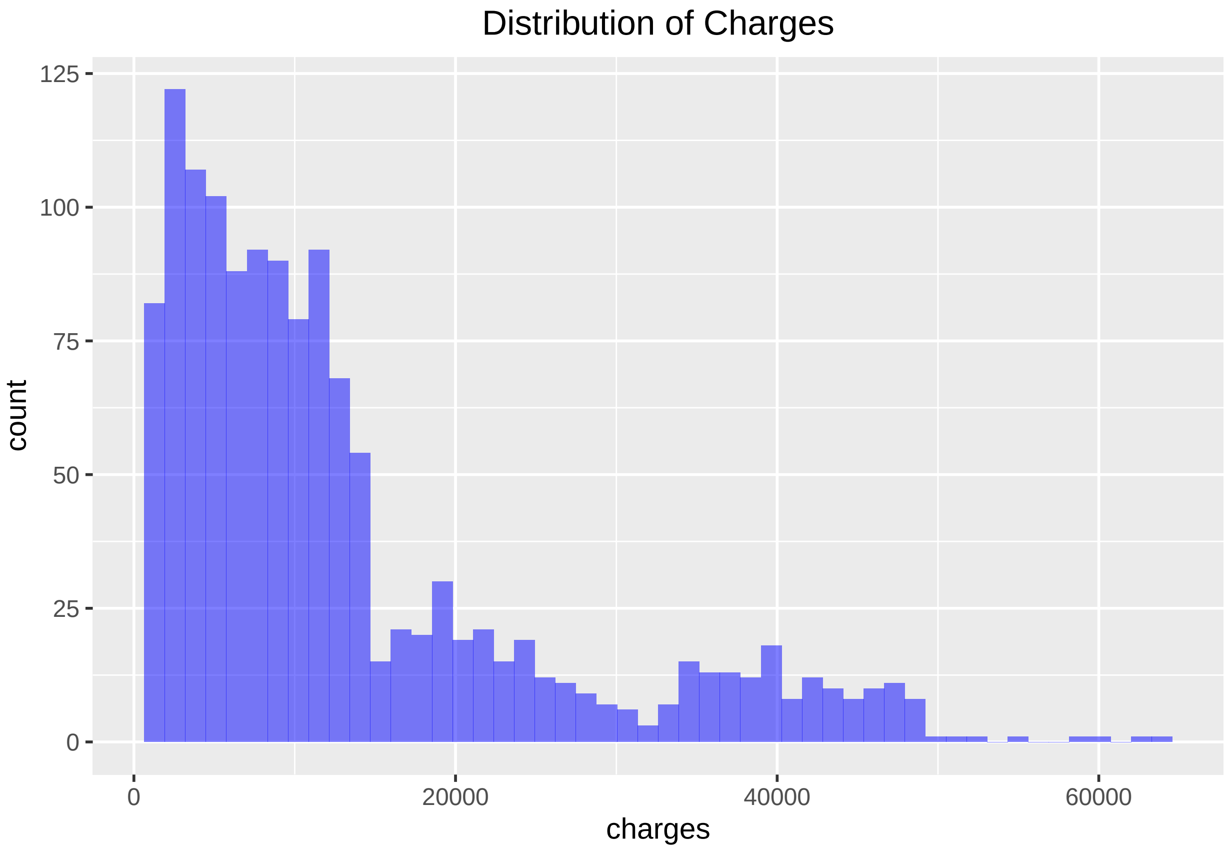
```
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

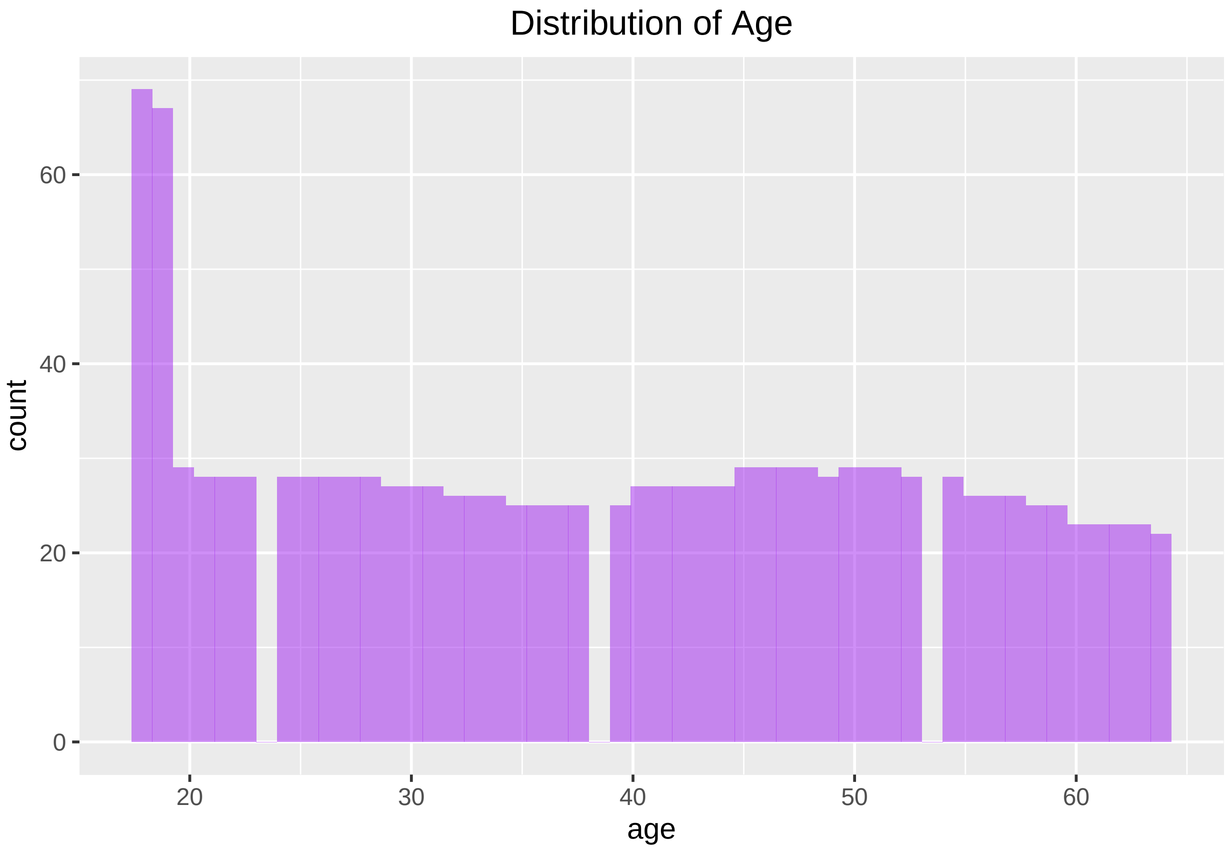
```
library(dplyr)
df<- distinct(dataset)#new data without duplicates
sum(duplicated(df))#unique values?
```

```
## [1] 0
```

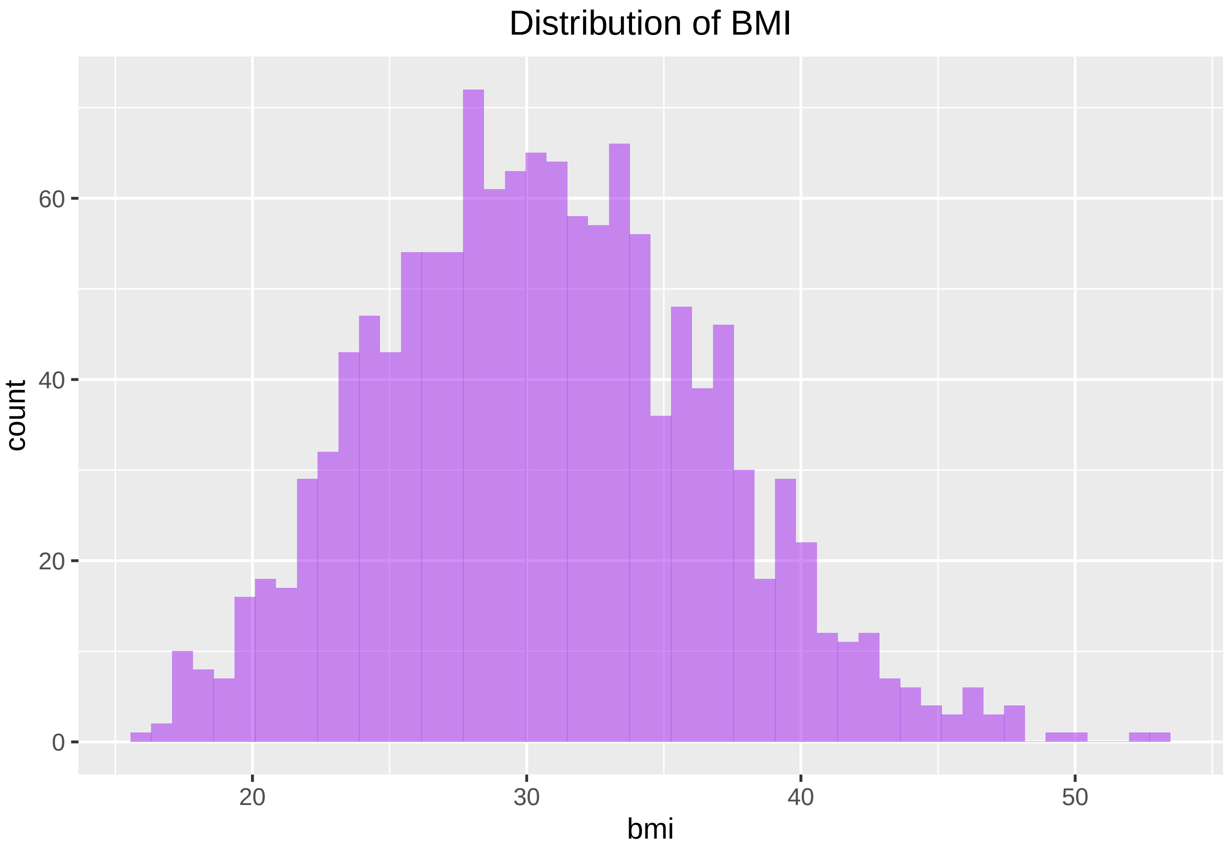
```
theme_set(theme_gray())
theme_update(plot.title = element_text(hjust = 0.5))

ggplot(df, aes(x = charges)) +
geom_histogram(bins =50,fill = "blue", alpha = 0.5) +
labs(title = "Distribution of Charges")
```



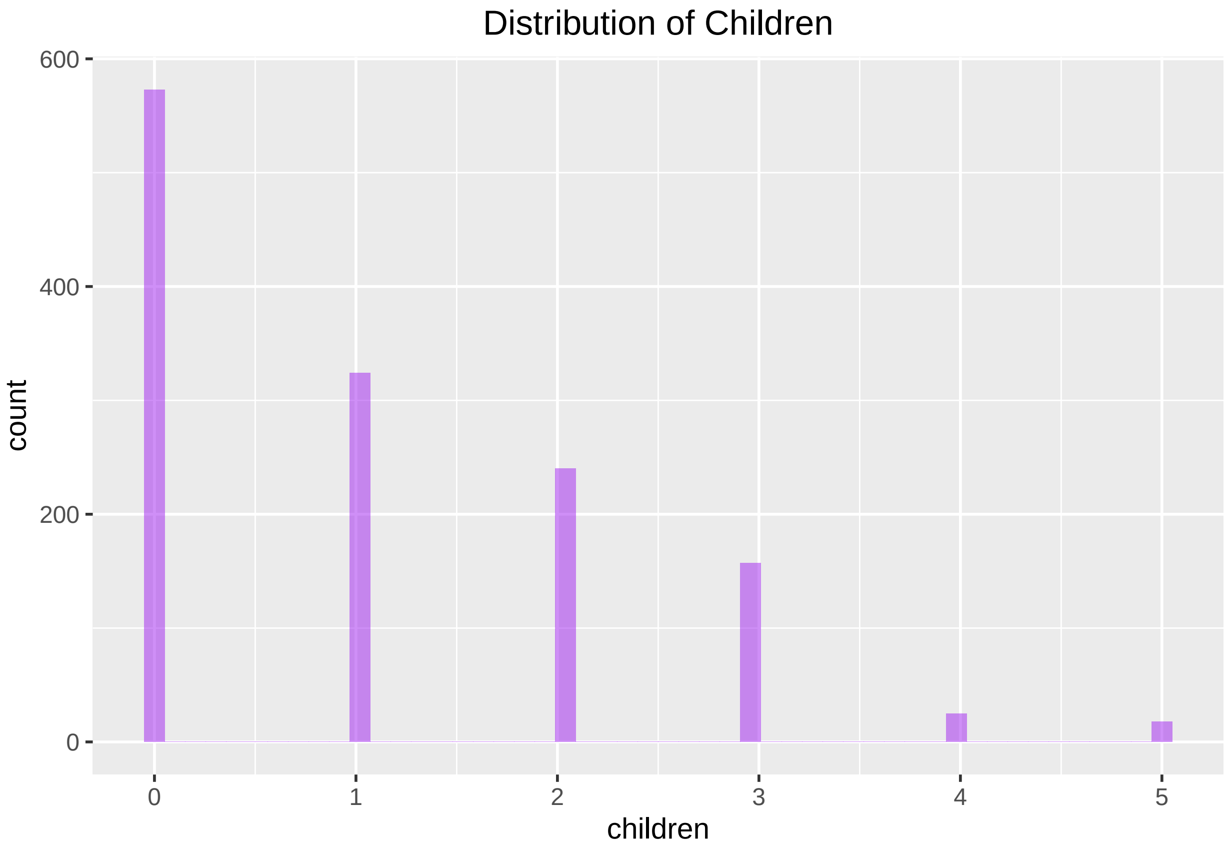
```
theme_set(theme_gray())
theme_update(plot.title = element_text(hjust = 0.5))

ggplot(df, aes(x = age)) +
geom_histogram(bins=50,fill = "purple", alpha = 0.5) +
labs(title = "Distribution of Age")
```



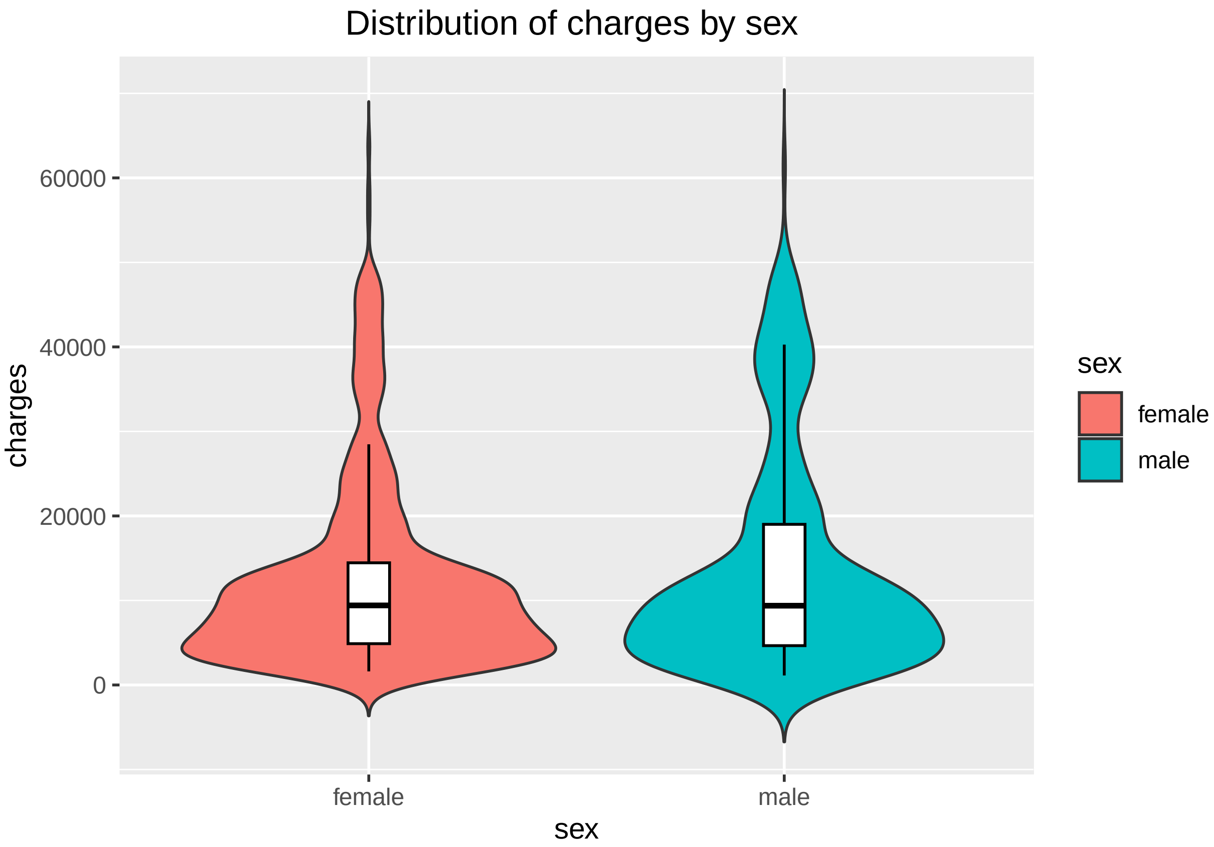
```
theme_set(theme_gray())
theme_update(plot.title = element_text(hjust = 0.5))

ggplot(df, aes(x = bmi)) +
geom_histogram(bins=50,fill = "purple",alpha=0.5) +
labs(title = "Distribution of BMI")
```



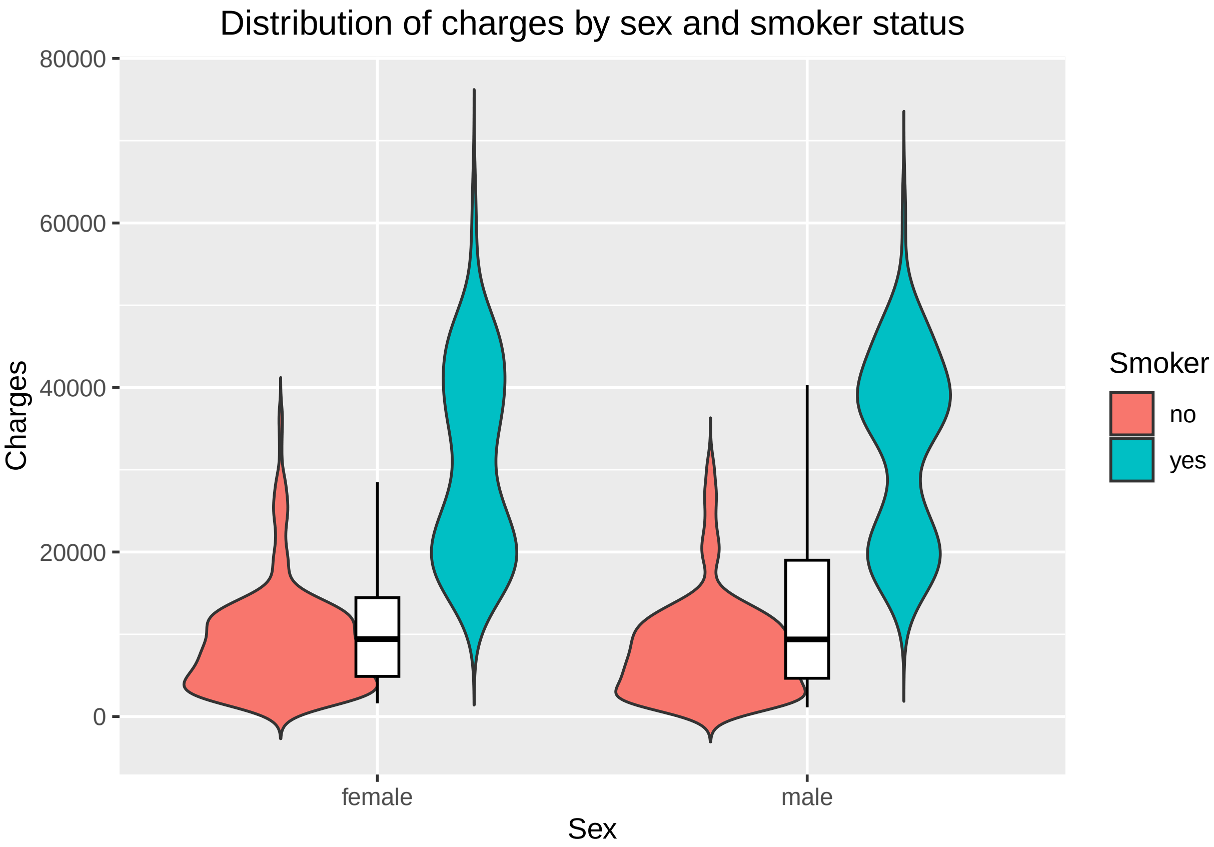
```
theme_set(theme_gray())
theme_update(plot.title = element_text(hjust = 0.5))

ggplot(df, aes(x = children)) +
geom_histogram(bins= 50, fill = "purple",alpha=0.5) +
labs(title = "Distribution of Children")
```



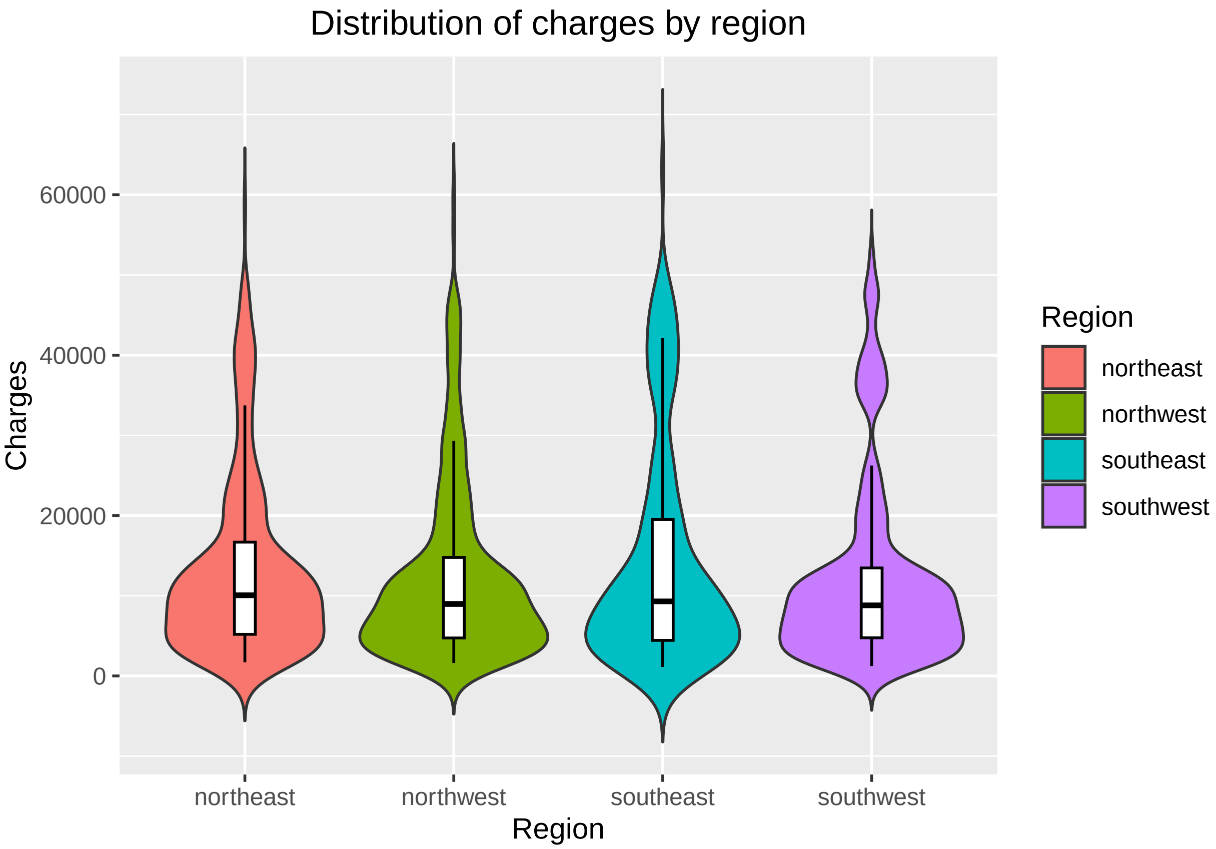
```
ggplot(df, aes(x=sex, y=charges, fill=sex)) +
  geom_violin(trim=FALSE) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape = NA) +
  labs(title="Distribution of charges by sex ",
       )
```



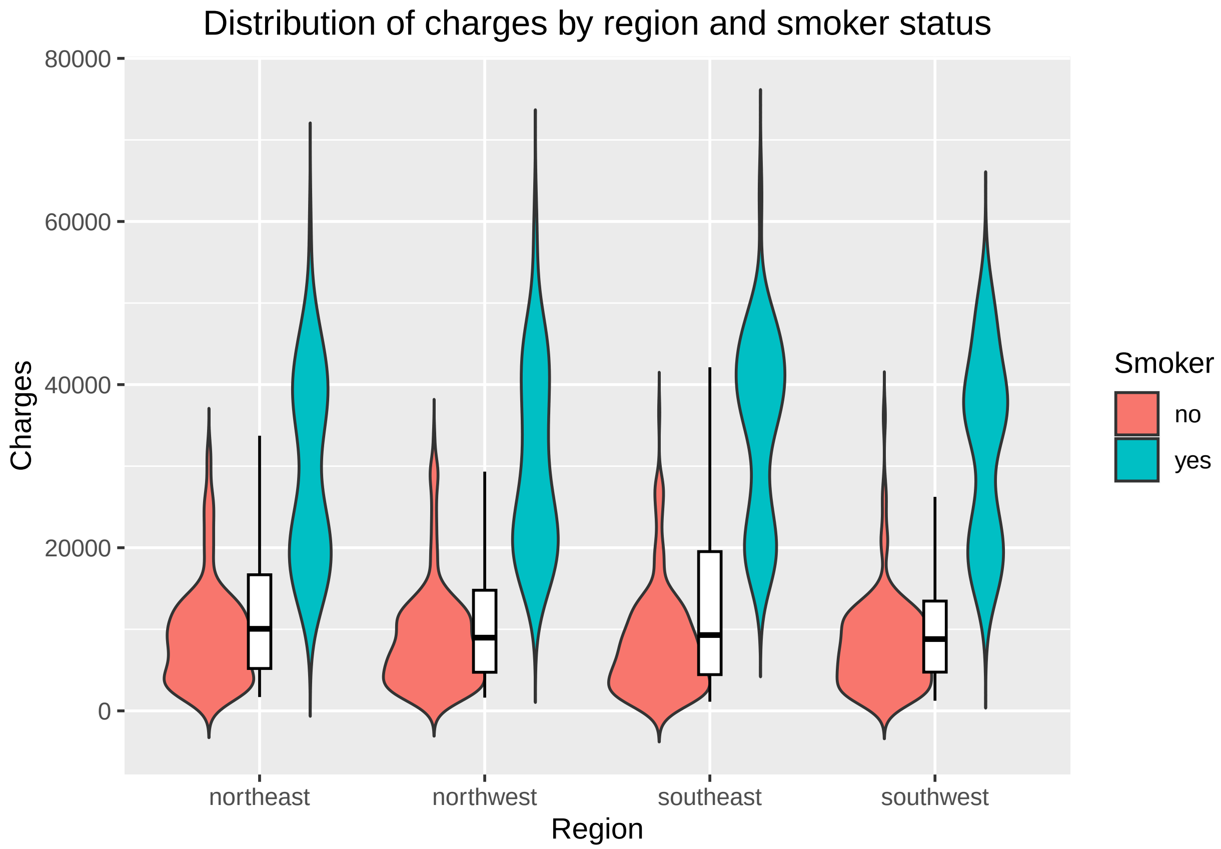
```
ggplot(df, aes(x=sex, y=charges, fill=smoker)) +
  geom_violin(trim=FALSE) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape = NA) +
  labs(title="Distribution of charges by sex and smoker status",
       x="Sex",
       y="Charges",
       fill="Smoker")
```



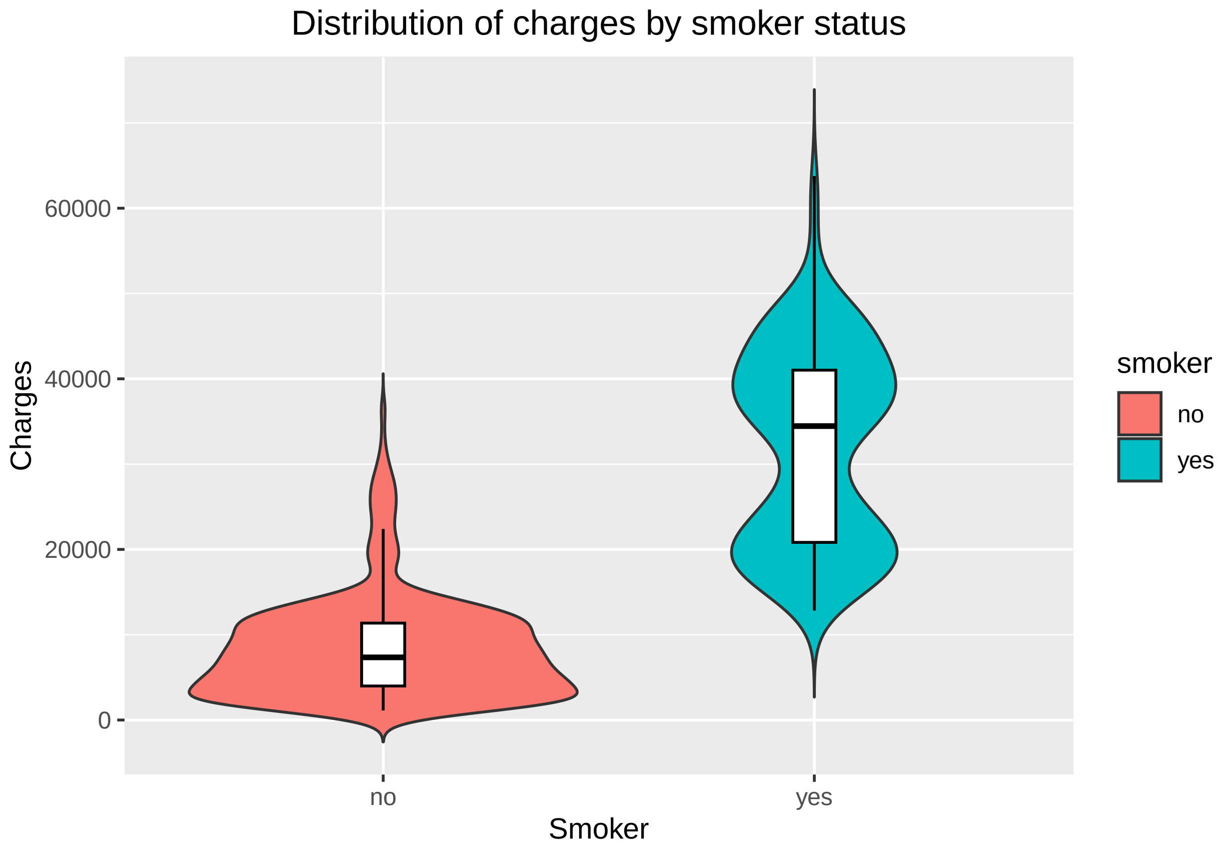
```
ggplot(df, aes(x=region, y=charges, fill=region)) +
  geom_violin(trim=FALSE) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape = NA) +
  labs(title="Distribution of charges by region",
       x="Region",
       y="Charges",
       fill="Region")
```



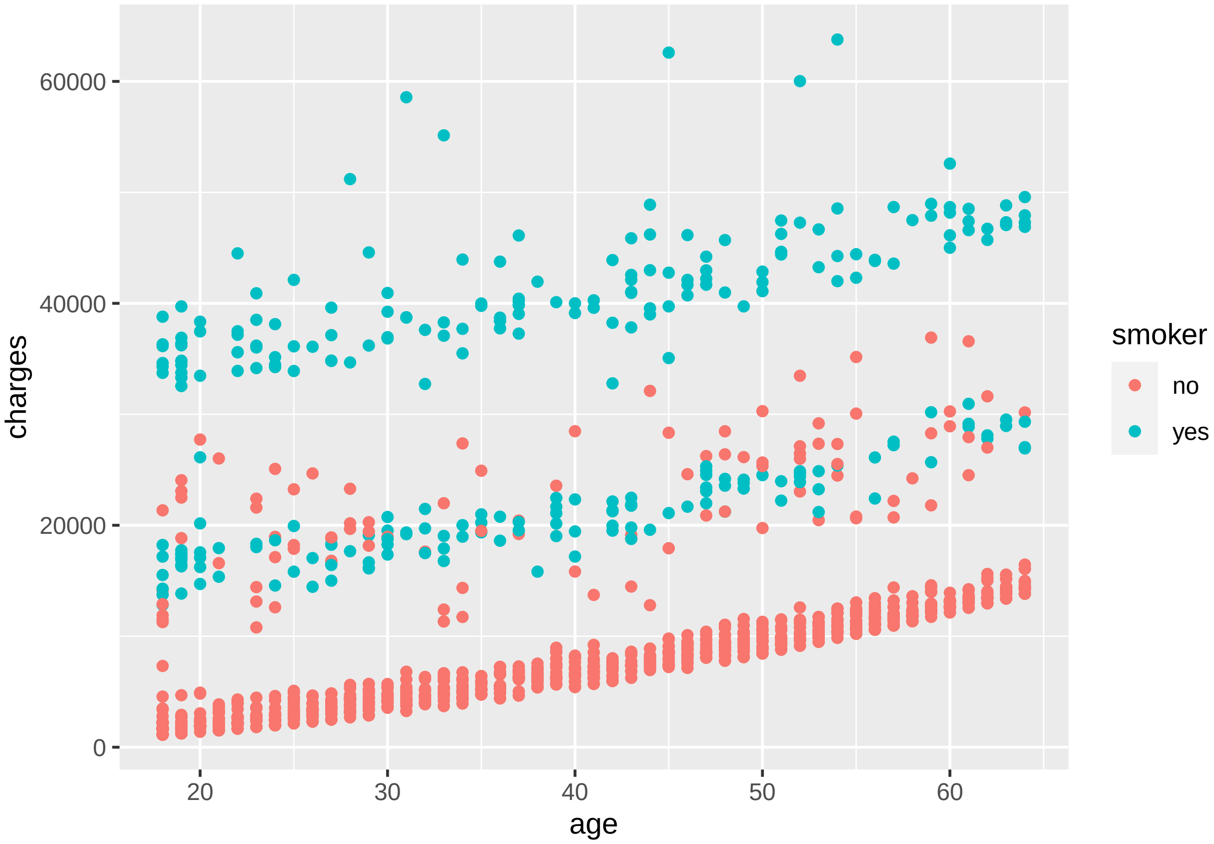
```
ggplot(df, aes(x=region, y=charges, fill=smoker)) +
  geom_violin(trim=FALSE) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape = NA) +
  labs(title="Distribution of charges by region and smoker status",
       x="Region",
       y="Charges",
       fill="Smoker")
```



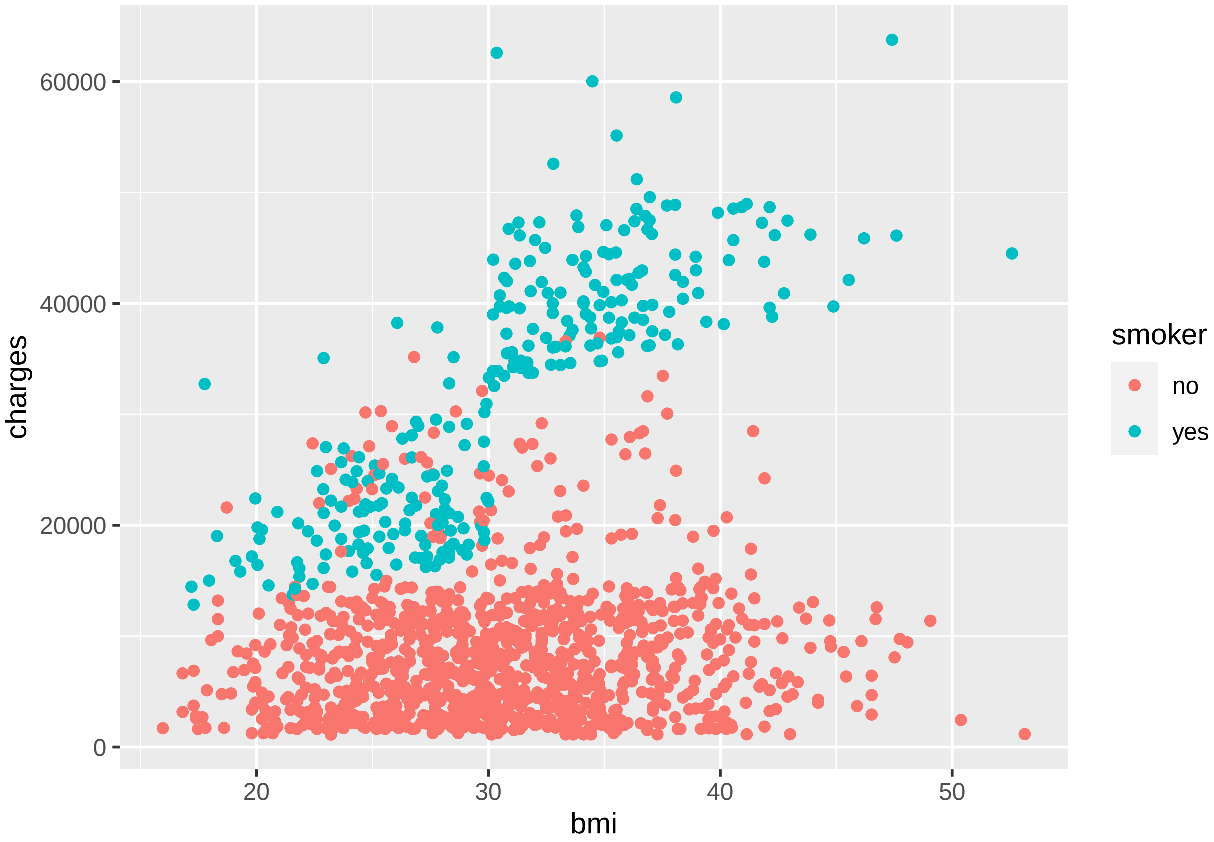
```
ggplot(df, aes(x=smoker, y=charges, fill=smoker)) +
  geom_violin(trim=FALSE) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape = NA) +
  labs(title="Distribution of charges by smoker status",
       x="Smoker",
       y="Charges")
```



```
ggplot(data = dataset, aes(x = age, y = charges, color = smoker)) +
  geom_point()
```



```
ggplot(data = dataset, aes(x = bmi, y = charges, color = smoker)) +
  geom_point()
```



## Our model

```
import lazypredict
import pandas as pd
import numpy as np
from lazypredict.Supervised import LazyRegressor
```

```
df = pd.read_csv("00-insurance.csv", index_col = 0).reset_index()
df.drop_duplicates(inplace=True)
df.head(2)
```

```
##    age     sex   bmi  children smoker     region   charges
## 0   19  female 27.90         0    yes  southwest 16884.92
## 1   18    male 33.77         1     no  southeast  1725.55
```

```
df['sex'] = df['sex'].map({'female':0,'male':1})
df['smoker'] = df['smoker'].map({'no':0,'yes':1})
df['region'] = df['region'].map({'northeast':1,'northwest':2,'southeast':3,'southwest':4})
df.head(2)
```

```
##    age  sex   bmi  children  smoker  region   charges
## 0   19    0 27.90         0       1       4 16884.92
## 1   18    1 33.77         1       0       3  1725.55
```

```
X = df.drop('charges',axis=1)
y = df['charges']
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=42)
```

```
from lazypredict.Supervised import LazyRegressor
clf = LazyRegressor(verbose=0)
models,predictions = clf.fit(x_train, x_test, y_train, y_test)
```

```
##   0%|          | 0/42 [00:00<?, ?it/s]  7%|7         | 3/42 [00:00<00:02, 18.72it/s] 19%|#9         |
```

```
models
```

```
##                                Adjusted R-Squared  ...  Time Taken
## Model                                              ...
## GradientBoostingRegressor                    0.90  ...        0.14
## HistGradientBoostingRegressor                0.89  ...        0.27
## LGBMRegressor                                0.88  ...        4.96
## BaggingRegressor                             0.88  ...        0.04
## RandomForestRegressor                        0.88  ...        0.33
## XGBRegressor                                 0.87  ...        1.16
## KNeighborsRegressor                          0.86  ...        0.04
## ExtraTreesRegressor                          0.85  ...        0.23
## PoissonRegressor                             0.83  ...        0.02
## AdaBoostRegressor                            0.83  ...        0.06
## Lars                                         0.80  ...        0.14
## TransformedTargetRegressor                   0.80  ...        0.01
## LinearRegression                             0.80  ...        0.01
## Lasso                                        0.80  ...        0.02
## LassoLars                                    0.80  ...        0.01
## Ridge                                        0.80  ...        0.03
## RidgeCV                                      0.80  ...        0.01
## SGDRegressor                                 0.80  ...        0.06
## BayesianRidge                                0.80  ...        0.06
## LassoLarsIC                                  0.80  ...        0.05
## LarsCV                                       0.80  ...        0.02
## LassoLarsCV                                  0.80  ...        0.02
## LassoCV                                      0.80  ...        0.05
## OrthogonalMatchingPursuitCV                  0.80  ...        0.01
## HuberRegressor                               0.78  ...        0.02
## PassiveAggressiveRegressor                   0.77  ...        0.03
## ExtraTreeRegressor                           0.77  ...        0.01
## DecisionTreeRegressor                        0.77  ...        0.01
## OrthogonalMatchingPursuit                    0.67  ...        0.01
## ElasticNet                                   0.66  ...        0.02
## RANSACRegressor                              0.55  ...        0.15
## TweedieRegressor                             0.54  ...        0.04
## GammaRegressor                               0.50  ...        0.03
## ElasticNetCV                                 0.11  ...        0.05
## DummyRegressor                              -0.03  ...        0.01
## NuSVR                                       -0.09  ...        0.14
## SVR                                         -0.16  ...        0.07
## QuantileRegressor                           -0.16  ...       65.85
## KernelRidge                                 -0.20  ...        0.42
## LinearSVR                                   -0.99  ...        0.02
## MLPRegressor                                -1.05  ...       20.47
## GaussianProcessRegressor                 -2695.41  ...        0.90
##
## [42 rows x 4 columns]
```

The result of lazypredict can be viewed in Notebook, positcloud does not allow the import of the code with that tool.

```
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
# Standardize the dataset
sc = StandardScaler()
x_train_std = sc.fit_transform(x_train)
x_test_std = sc.transform(x_test)

# Hyperparameters for GradientBoostingRegressor
#
gbr_params = {'n_estimators': 1000,
          'max_depth': 3,
          'min_samples_split': 5,
          'learning_rate': 0.01,
          'loss': 'absolute_error'}
```

```
# Create an instance of gradient boosting regressor
#
gbr = GradientBoostingRegressor(**gbr_params)
#
# Fit the model
#
gbr.fit(x_train_std, y_train)
```

```
## GradientBoostingRegressor(learning_rate=0.01, loss='absolute_error',
##                           min_samples_split=5, n_estimators=1000)
```

```
# Print Coefficient of determination R^2
#
print("Model Accuracy: %.3f" % gbr.score(x_test_std, y_test))
#
# Create the mean squared error
#
```

```
## Model Accuracy: 0.809
```

```
mse = mean_squared_error(y_test, gbr.predict(x_test_std))
print("The mean squared error (MSE) on test set: {:.4f}".format(mse))
#
```

```
## The mean squared error (MSE) on test set: 35059926.3119
```

Best metrics are given by GradientBoostingRegressor, our response variable does not have a normal distribution, nor is the relationship with the response variables entirely linear. We could fit the model with interval prediction.