HOUSING PREDICTION

Kayla Lopilato

Nicholas Scarth

Makhi Carr

PURPOSE

Using machine learning, predict the price of housing with 90% accuracy

DATA SOURCE

Mock dataset from Kaggle

“Housing Prices Dataset” by M Yasser H

https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

DATA CLEANING

When initially cleaning the dataset, we removed the columns guest room, preferred area, and furnishing status due to the subjectivity of the columns.

```
# Imports
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
# Scikit Learn Imports
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn import metrics
# Tensorflow Imports
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

```
# Import Data
data_import = pd.read_csv(".//house_predictions.csv")
print(data_import)
```

Prepare Data

```
# Create pandas dataframe
house_info = pd.DataFrame(data_import)
house_info.head()
```

ooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
4	2	3	yes	no	no	no	yes	2	yes	furnished
4	4	4	yes	no	no	no	yes	3	no	furnished
3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	1	2	yes	yes	yes	no	yes	2	no	furnished

```
# Drop unnecessary columns
house_df = house_info.drop(columns=["guestroom", "prefarea", "furnishingstatus"], axis=1)
house_df.head()
```

	price	area	bedrooms	bathrooms	stories	mainroad	basement	hotwaterheating	airconditioning	parking
0	13300000	7420	4	2	3	yes	no	no	yes	2
1	12250000	8960	4	4	4	yes	no	no	yes	3
2	12250000	9960	3	2	2	yes	yes	no	no	2
3	12215000	7500	4	2	2	yes	yes	no	yes	3
4	11410000	7420	4	1	2	yes	yes	no	yes	2

```
# Split X and Y
y = converted_house_df["price"]

x = converted_house_df.copy()
x = x.drop(columns="price")
```

```
# Train Test Split
x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=4)
```

```
# Scale numeric data
scaler = StandardScaler().fit(x_train)
scaled_x_train = scaler.transform(x_train)
scaled_x_test = scaler.transform(x_test)
```

MODEL #1

Linear regression model scored 61.0%

MODEL #2

Random Forest model scored 52.5%

MODEL #3

Gradient Boosting model scored 43.7%

MODEL #4

Neural Network model scored 61.9%

```
# Alternate model - Gradient Boosting Regression
gbr_model = GradientBoostingRegressor(n_estimators=1000, random_state=4)

gbr_model.fit(scaled_x_train, y_train)
gbr_model_predictions = gbr_model.predict(scaled_x_test)
```

```
# Score model
score_model(y_test, gbr_model_predictions)
```

```
R2 score: 0.43731582754579  Mean Square Error: 1807243238507.416  Root Mean Square Error: 1344337.
4719568803
```

MORE DATA CLEANING…

Since the four models did not perform as expected, we decided to see if removing additional columns would help the models perform better.

```
# Reduce columns
reduced_x = x.copy()
reduced_x = reduced_x[['area', 'bedrooms', 'bathrooms', 'stories', 'basement', 'airco
```

```
# Split and Scale reduced df
x_train_2, x_test_2, y_train_2, y_test_2 = train_test_split(reduced_x, y, random_stat

scaler_2 = StandardScaler().fit(x_train_2)
scaled_x_train_2 = scaler_2.transform(x_train_2)
scaled_x_test_2 = scaler_2.transform(x_test_2)
```

MODEL #5

Linear Regression model scored 59.3%

MODEL #6

Random Forest model scored 54.1%

MODEL #7

Gradient Boosting model scored 41.8%

MODEL #8

Neural Network model scored 59.8%

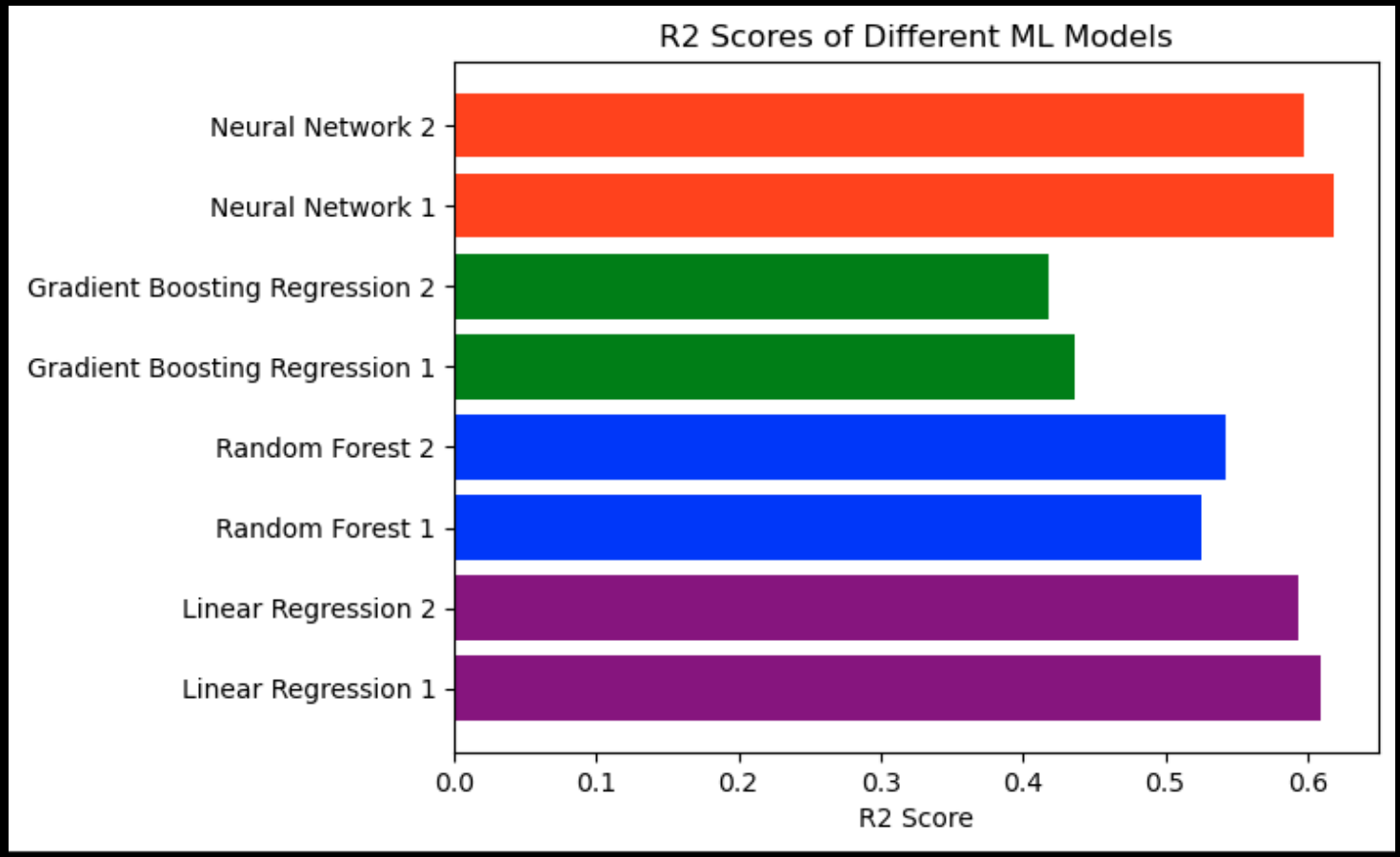
```
# GBR model 2
gbr_model_2 = GradientBoostingRegressor(n_estimators=1000, random_state=4)

gbr_model_2.fit(scaled_x_train_2, y_train_2)
gbr_model_2_predictions = gbr_model_2.predict(scaled_x_test_2)
```

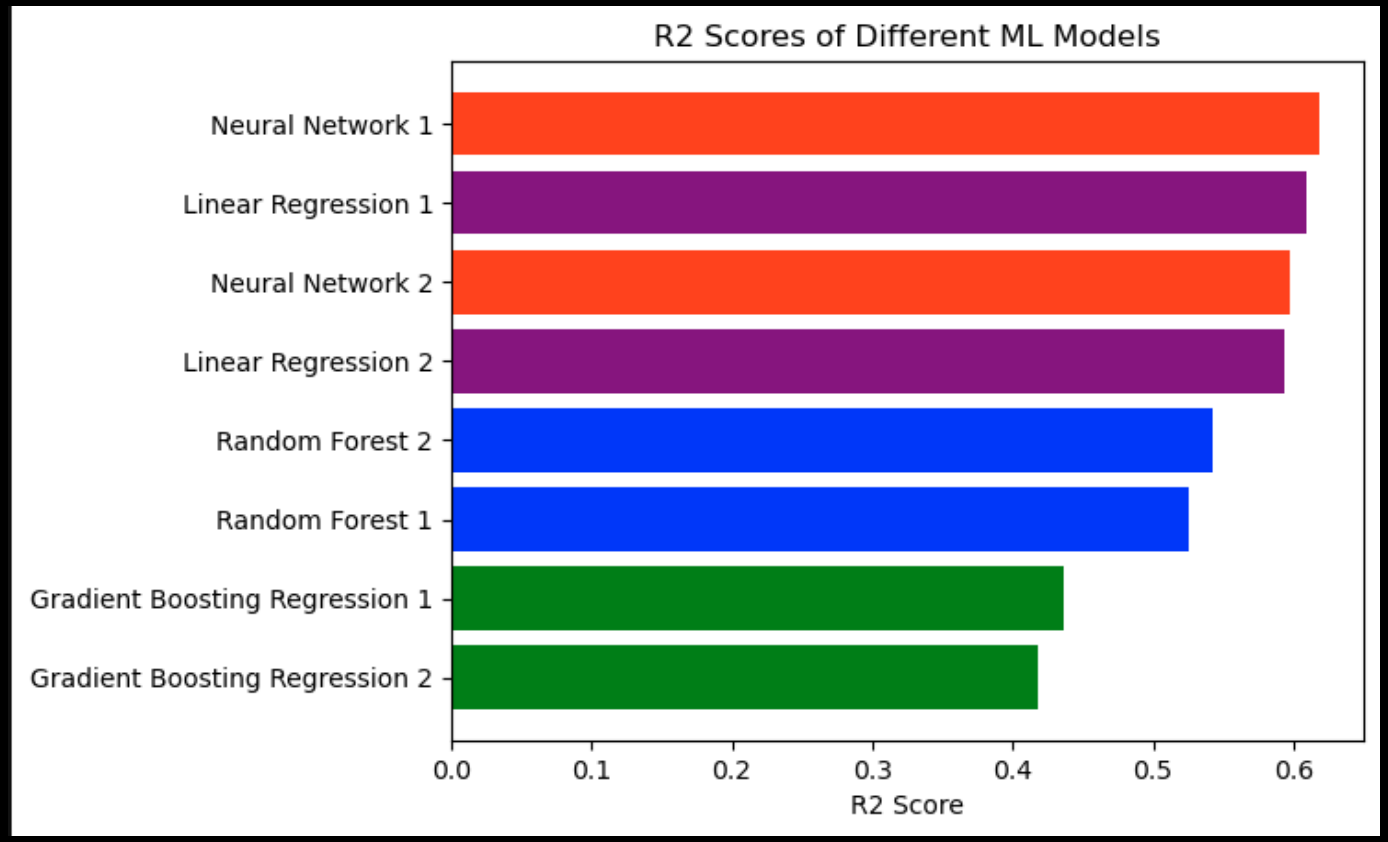
```
# Score model
score_model(y_test_2, gbr_model_2_predictions)
```

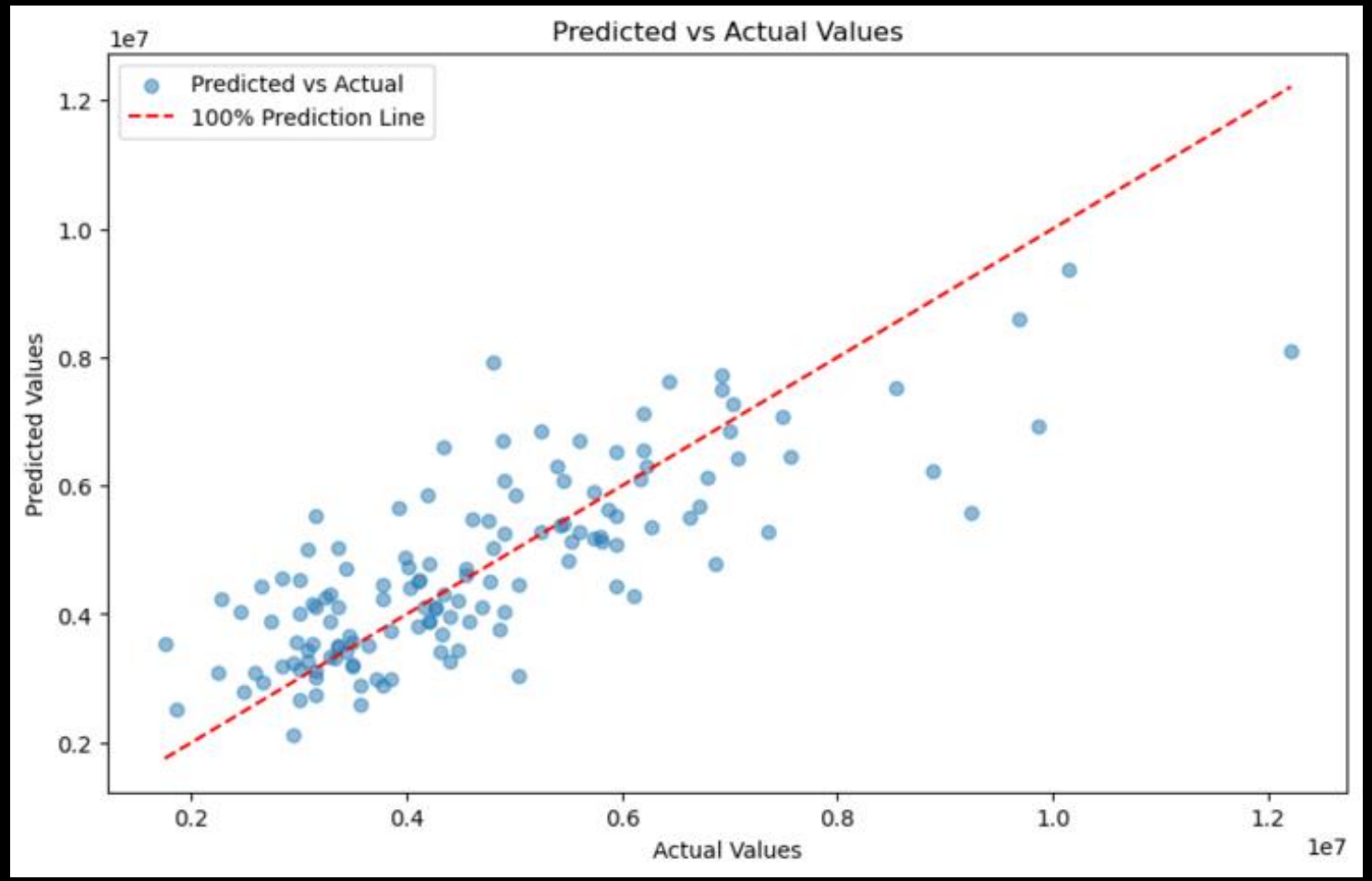
```
R2 score: 0.417923847057344  Mean Square Error: 1869526891282.213  Root Mean Square Er
ror: 1367306.4364955695
```

REDUCING THE COLUMNS REDUCED THE R2 SCORE.



WHEN COMPARING THE MODELS, NEURAL NETWORKS AND LINEAR REGRESSION PERFORM BEST IN BOTH DATASETS WHILE GRADIENT BOOSTING DOESN'T APPEAR TO BE THE BEST OPTION.





EVEN THOUGH THE MODELS DIDN'T PERFORM AS EXPECTED IN BOTH DATASETS, THEY STILL ACHIEVED A PREDICTION WITH FEW OUTLIERS.

LIMITATIONS

ARTIFICIAL DATASET

545 ROWS – MORE DATA MIGHT HAVE ACHIEVED BETTER RESULTS

SUMMARY

- After removing subjective columns, we attempted four models to reach accurate predictions.
- After these four models didn't reach the accuracy goal of 90%, two more columns were removed, and the models were retried.
- Even after removing additional columns, the models performed lower than the initial models.
- Of the four models, the neural network model performed the best.

THANK YOU