Client Report - Can you predict that?

Course DE 250
Conner Crook

Elevator pitch

When picking a home, the amount of livable square footage a home has is the most important feature that people look for. Even though this is the most important feature, many other features have almost as high importance.

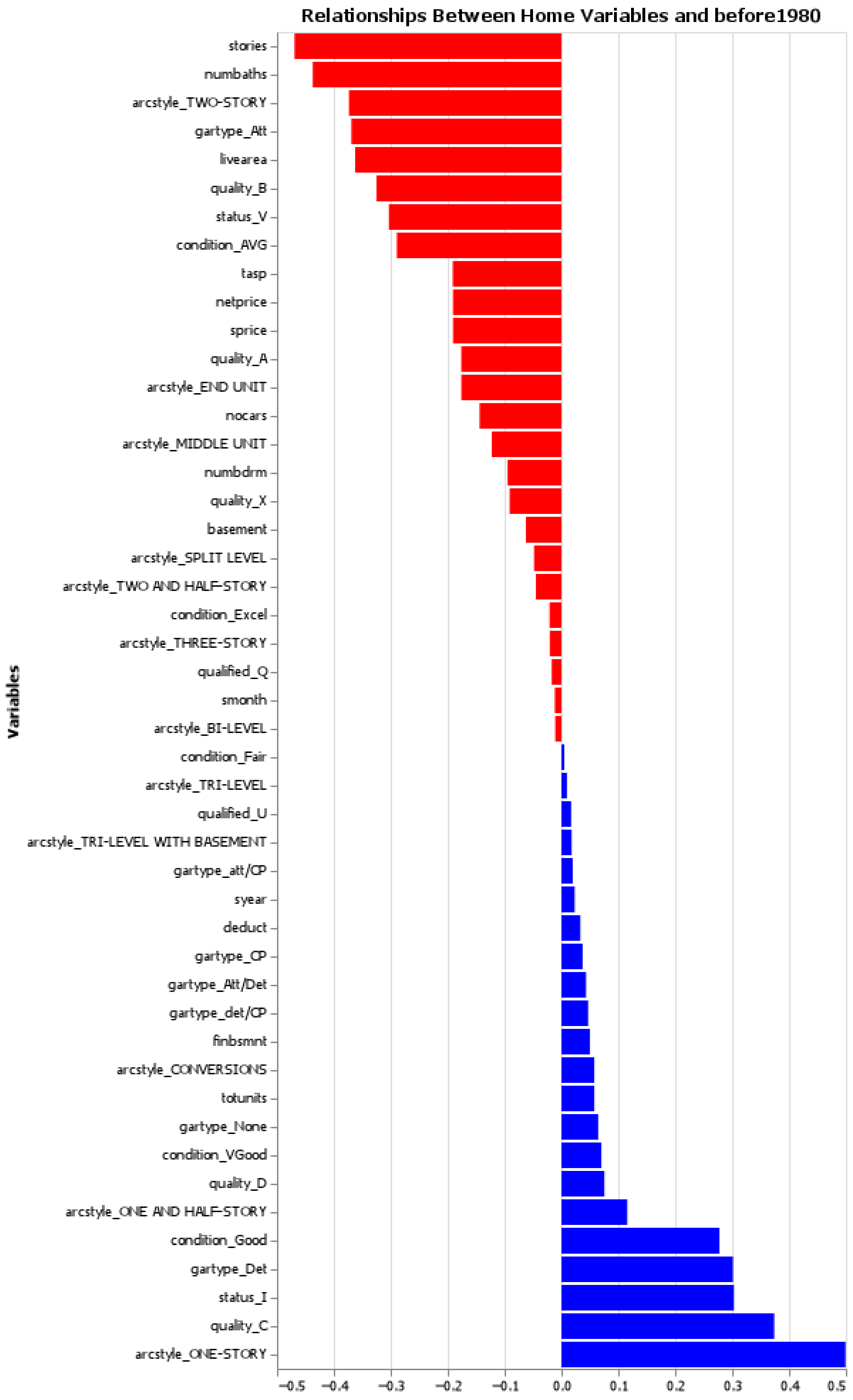
GRAND QUESTION 1

Create 2-3 charts that evaluate potential relationships between the home variables and before1980.

_From these two charts, we see that stories has the highest negative correlation at almost -.05. arcstyle_ONE-STORY has the largest positive correlation at 0.5. The average correlation is in between 0 and -.1.

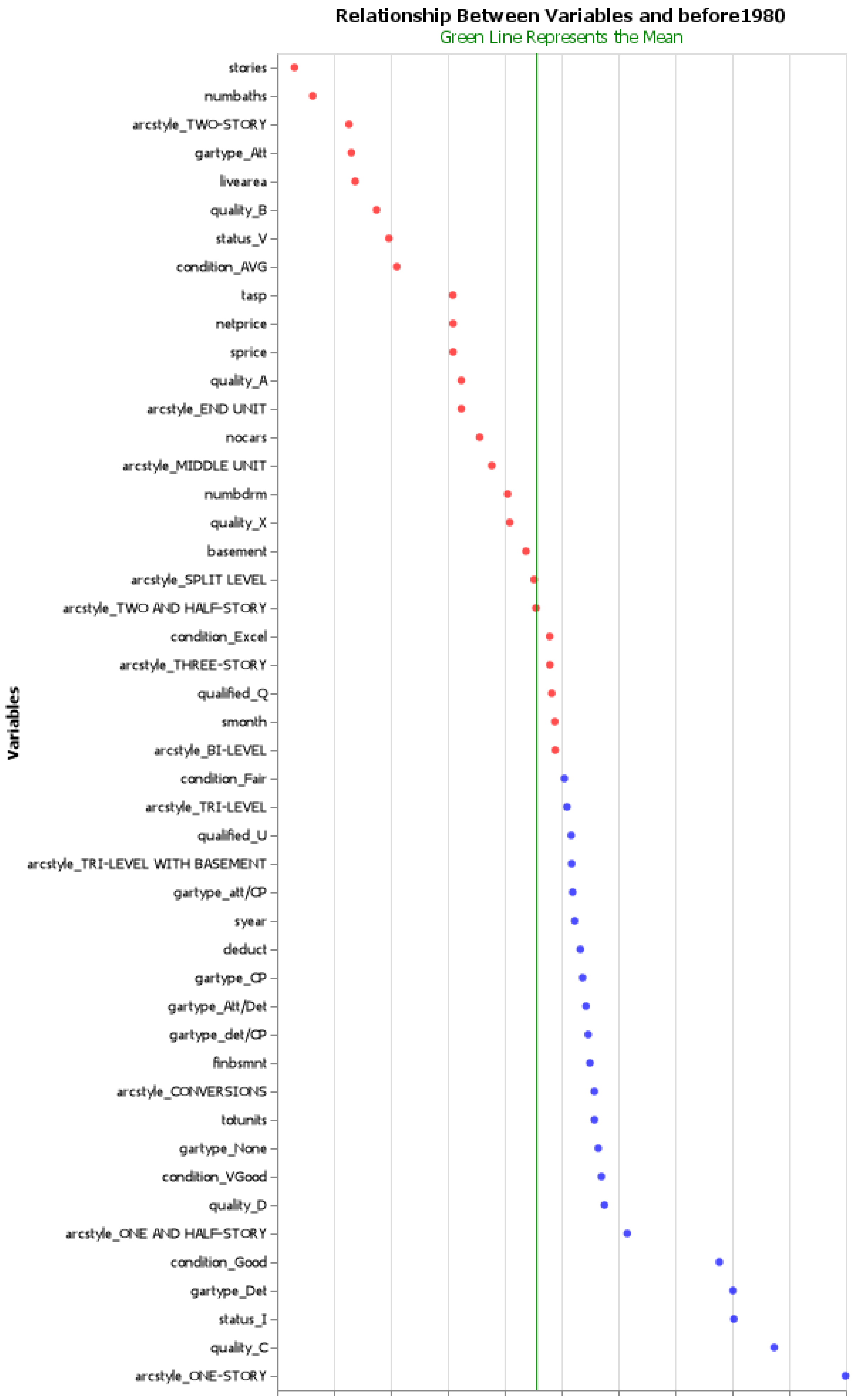
TECHNICAL DETAILS

```
variable_correlations = dwellings_ml.corr(method = 'pearson').before1980.sort_values()

variable_correlations.drop([
  'before1980',
  'abstrprd',
  'yrbuilt'], inplace = True)

variable_correlations = pd.DataFrame(variable_correlations.reset_index())

#%%
# Make the chart
correlation_chart_one = (alt
  .Chart(variable_correlations)
  .mark_bar()
  .encode(
    x = alt.X(
      'before1980',
      axis = alt.Axis(title = 'Correlation')),
    y = alt.Y(
      'index:N',
      axis = alt.Axis(title = 'Variables'),
      sort = None),
    color=alt.condition(
        alt.datum.before1980 > 0,
        alt.value("blue"),
        alt.value("red")
    )
  )
  .properties(
    title = 'Relationships Between Home Variables and before1980'
  )
  )
```



Relationships Between Home Variables and before1980
Variables
stories
numbaths
arcstyle_TWO-STORY
gartype_Att
livearea
quality_B
status_V
condition_AVG
tasp
netprice
sprice
quality_A
arcstyle_END UNIT
nocars
arcstyle_MIDDLE UNIT
numbdrm
quality_X
basement
arcstyle_SPLIT LEVEL
arcstyle_TWO AND HALF-STORY
condition_Excel
arcstyle_THREE-STORY
qualified_Q
smonth
arcstyle_BI-LEVEL
condition_Fair
arcstyle_TRI-LEVEL
qualified_U
arcstyle_TRI-LEVEL WITH BASEMENT
gartype_att/CP
syear
deduct
gartype_CP
gartype_Att/Det
gartype_det/CP
finbsmnt
arcstyle_CONVERSIONS
totunits
gartype_None
condition_VGood
quality_D
arcstyle_ONE AND HALF-STORY
condition_Good
gartype_Det
status_I
quality_C
arcstyle_ONE-STORY
−0.5
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
0.5

```
correlation_chart_two = (alt.Chart(variable_correlations)
.mark_circle()
.encode(
  x = alt.X('before1980',
    axis = alt.Axis(title = "Correlation")),
  y = alt.Y('index:N',
    axis = alt.Axis(title = "Variables"), sort = None),
  color = alt.condition(
    alt.datum.before1980 > 0,
    alt.value("blue"),
    alt.value("red")
  )
))

zero = (alt.Chart(variable_correlations)
  .mark_rule()
  .encode(
    x = alt.X("mean(before1980):Q"),
    color = alt.value('green')
  )
  .properties(
    title={
      "text" : ["Relationship Between Variables and before1980"],
      "subtitle" : ["Green Line Represents the Mean"],
      "color": "black",
      "subtitleColor": "green"
    }
  ))
```


Relationship Between Variables and before1980
Green Line Represents the Mean
Variables
stories
numbaths
arcstyle_TWO-STORY
gartype_Att
livearea
quality_B
status_V
condition_AVG
tasp
netprice
sprice
quality_A
arcstyle_END UNIT
nocars
arcstyle_MIDDLE UNIT
numbdrm
quality_X
basement
arcstyle_SPLIT LEVEL
arcstyle_TWO AND HALF-STORY
condition_Excel
arcstyle_THREE-STORY
qualified_Q
smonth
arcstyle_BI-LEVEL
condition_Fair
arcstyle_TRI-LEVEL
qualified_U
arcstyle_TRI-LEVEL WITH BASEMENT
gartype_att/CP
syear
deduct
gartype_CP
gartype_Att/Det
gartype_det/CP
finbsmnt
arcstyle_CONVERSIONS
totunits
gartype_None
condition_VGood
quality_D
arcstyle_ONE AND HALF-STORY
condition_Good
gartype_Det
status_I
quality_C
arcstyle_ONE-STORY

GRAND QUESTION 2

Can you build a classification model (before or after 1980) that has at least 90% accuracy for the state of Colorado to use (explain your model choice and which models you tried)?

For my model, I took out before1980 and yrbuilt because they are too related to the target, 1980. I also took out parcel because I don't think it classifies as a feature and I removed abstrprd because we are unsure what that column is. I used the RandomForestClassifier from sklearn. The model results show us that the square footage of the home that is livable (livearea) is the most important with a value of 0.0894899.

TECHNICAL DETAILS

```
features = dwellings_ml.drop(dwellings_ml.filter(regex = 'before1980|abstrprd|yrbuilt|parcel').c
target = dwellings_ml.filter(regex = "before1980")
feature_train, feature_test, target_train, target_test = train_test_split(features, target, test

#%%
# Use the RandomForestclassifier
classifier = RandomForestClassifier()
classifier.fit(feature_train, target_train)
#%%
# Predict using classifier
targets_predicted = classifier.predict(feature_test)

#%%
#Create Table
features_importance = pd.DataFrame(
    {'feature': feature_train.columns,
    'importance': classifier.feature_importances_}).sort_values('importance', ascending = False)


print(features_importance
    .head(20)
    .to_markdown(index = False))
```

replace the table below with your table

feature	importance
livearea	0.0894899

feature	importance
stories	0.08655
arcstyle_ONE-STORY	0.0819053
netprice	0.0610115
tasp	0.0594857
numbaths	0.0559113
sprice	0.0558688
gartype_Att	0.049887
basement	0.0425954
quality_C	0.0399004
nocars	0.0308497
quality_B	0.0287608
condition_AVG	0.0284553
numbdrm	0.0274964
finbsmnt	0.026973
smonth	0.0264871
gartype_Det	0.0253049
arcstyle_TWO-STORY	0.0216931
condition_Good	0.0183584
deduct	0.0176156

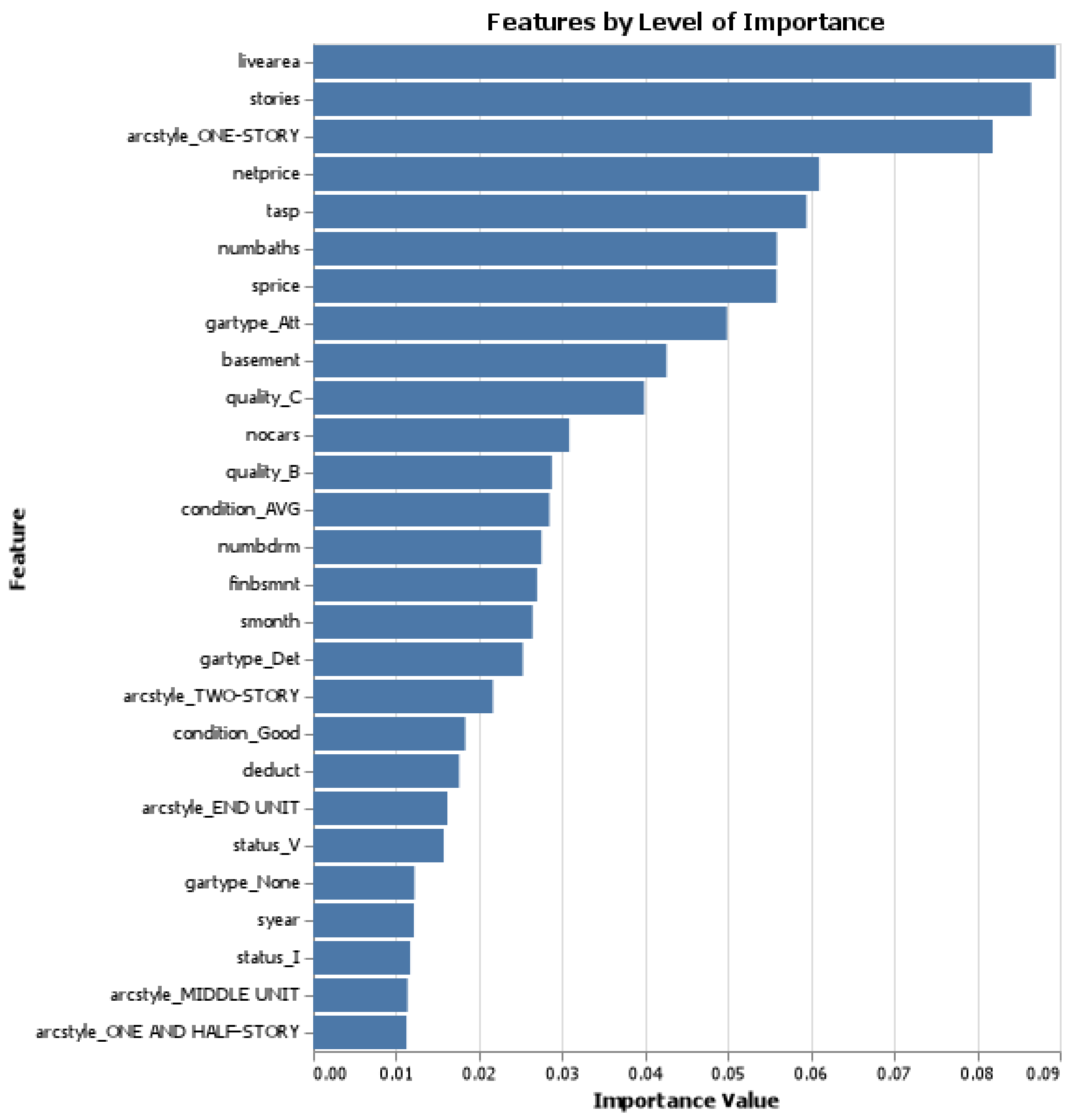
GRAND QUESTION 3

Will you justify your classification model by detailing the most important features in your model (a chart and a description are a must)?

The most important feature in the model is livearea, or the square footage of the home that is liveable. Even the highest had a small value of 0.0894899. The next 3 most important features in the model are *the number of stories (stories), arcstyle_ONE-STORY, and netprice. The chart shows only those that had a value above 0.007.*

TECHNICAL DETAILS

```
(alt.Chart(features_importance.query('importance > .007'))
    .encode(
        alt.X('importance'),
        alt.Y('feature', sort = '-x'))
    .mark_bar())
```



GRAND QUESTION 4

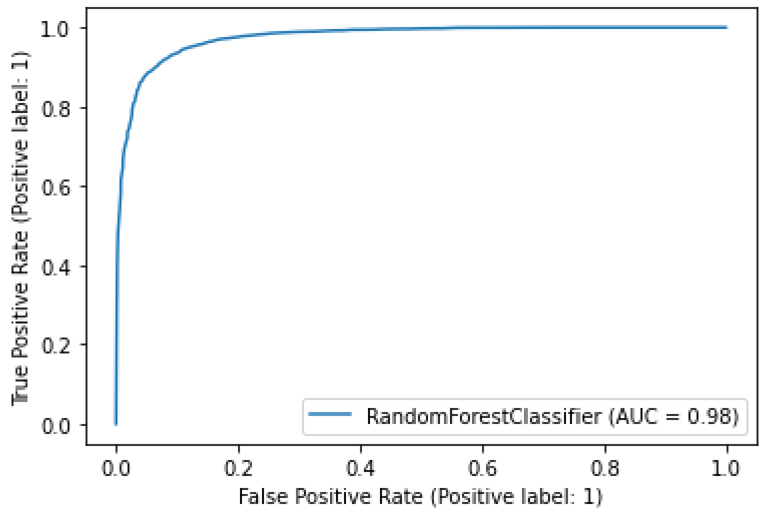
Can you describe the quality of your classification model using 2-3 evaluation metrics? You need to provide an interpretation of each evaluation metric when you provide the value.

The three evaluation metrics that I used are the accuracy_score, classification report, and the roc curve plot. The accuracy score gave a score of 0.924 meaning that it is 92.4% correct. The classification report and roc curve plot are shown below. The classification report shows that the model had 0.89 True Negatives (0) and 0.94 True Positives (1). The model also had above .90 in recall for both true negative and true positive. The chart below of the roc curve plot shows that the model has a AUC of 0.98.

TECHNICAL DETAILS

```
#Test Accuracy
accuracy_score(target_test, targets_predicted)
#%%
# Print classification report and show roc_curve

# 1 = True positive and 0 = True Negative
print(metrics.classification_report(targets_predicted, target_test))

metrics.plot_roc_curve(classifier, feature_test, target_test)
```

insert your chart png here



```
print(metrics.classification_report(targets_predicted, target_test))
```

```
            precision    recall  f1-score   support

       0       0.89      0.91      0.90      2580
       1       0.94      0.93      0.94      4294
accuracy                           0.92      6874
```

	precision	recall	f1-score	support

	precision	recall	f1-score	support
macro avg	0.92	0.92	0.92	6874
weighted avg	0.92	0.92	0.92	6874

APPENDIX A (PYTHON CODE)

```
#%%
import pandas as pd
import numpy as np
import altair as alt
import seaborn as sns
#%%
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn import metrics
#from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
#%%
dwellings_ml = pd.read_csv('https://github.com/byuidatascience/data4dwellings/raw/master/data-ra
dwellings_ml
#%%
##GRAND QUESTION 1
"""
Create 2-3 charts that evaluate potential relationships between the home variables and before198

Finding Correlation between all the variables adn before1980

Dropping some of the variables that are not helpful: before1980, abstrprd
"""
variable_correlations = dwellings_ml.corr(method = 'pearson').before1980.sort_values()

variable_correlations.drop([
  'before1980',
  'abstrprd',
  'yrbuilt'], inplace = True)

variable_correlations = pd.DataFrame(variable_correlations.reset_index())

#%%
# Make the chart
correlation_chart_one = (alt
  .Chart(variable_correlations)
  .mark_bar()
  .encode(
    x = alt.X(
      'before1980',
      axis = alt.Axis(title = 'Correlation')),
    y = alt.Y(
      'index:N',
      axis = alt.Axis(title = 'Variables'),
      sort = None),
    color=alt.condition(
        alt.datum.before1980 > 0,
        alt.value("blue"),
        alt.value("red")
      )
    )
    .properties(
      title = 'Relationships Between Home Variables and before1980'
    )
    )

correlation_chart_one.save('predict_chart_one.png')
#%%
# Create second chart
correlation_chart_two = (alt.Chart(variable_correlations)
.mark_circle()
.encode(
  x = alt.X('before1980',
    axis = alt.Axis(title = "Correlation")),
  y = alt.Y('index:N',
    axis = alt.Axis(title = "Variables"), sort = None),
  color = alt.condition(
    alt.datum.before1980 > 0,
    alt.value("blue"),
    alt.value("red")
  )
))

zero = (alt.Chart(variable_correlations)
  .mark_rule()
  .encode(
    x = alt.X("mean(before1980):Q"),
    color = alt.value('green')
  )
  .properties(
    title={
      "text" : ["Relationship Between Variables and before1980"],
      "subtitle" : ["Green Line Represents the Mean"],
      "color": "black",
      "subtitleColor": "green"
    }
  ))

predict_chart_two = correlation_chart_two + zero

predict_chart_two.save('predict_chart_two.png')
#%%
##GRAND QUESTION 2
"""
Can you build a classification model (before or after 1980) that has at least 90% accuracy
for the state of Colorado to use (explain your model choice and which models you tried)?
"""

#Seperate the data into features and targets
features = dwellings_ml.drop(dwellings_ml.filter(regex = 'before1980|abstrprd|yrbuilt|parcel').c
target = dwellings_ml.filter(regex = "before1980")
feature_train, feature_test, target_train, target_test = train_test_split(features, target, test

#%%
# Use the RandomForestclassifier
classifier = RandomForestClassifier()
classifier.fit(feature_train, target_train)
#%%
# Predict using classifier
targets_predicted = classifier.predict(feature_test)

#%%
## GRAND QUESTION 3
features_importance = pd.DataFrame(
    {'feature': feature_train.columns,
    'importance': classifier.feature_importances_}).sort_values('importance', ascending = False)

feature_chart = (alt.Chart(features_importance.query('importance > .007'))
    .encode(
        alt.X('importance', title = "Importance Value"),
        alt.Y('feature', title = "Feature" ,sort = '-x'))
    .mark_bar()
    .properties(
      title = "Features by Level of Importance"
    ))

feature_chart.save('feature_chart.png')
#%%
## GRAND QUESTION 4
#Test Accuracy
accuracy_score(target_test, targets_predicted)
#%%
# Print classification report and show roc_curve

# 1 = True positive and 0 = True Negative
print(metrics.classification_report(targets_predicted, target_test))

metrics.plot_roc_curve(classifier, feature_test, target_test)
#%%
print(features_importance
    .head(20)
    .to_markdown(index = False))
#%%
```