Prediction Using Machine Learning

Austin Wilson

Elevator pitch

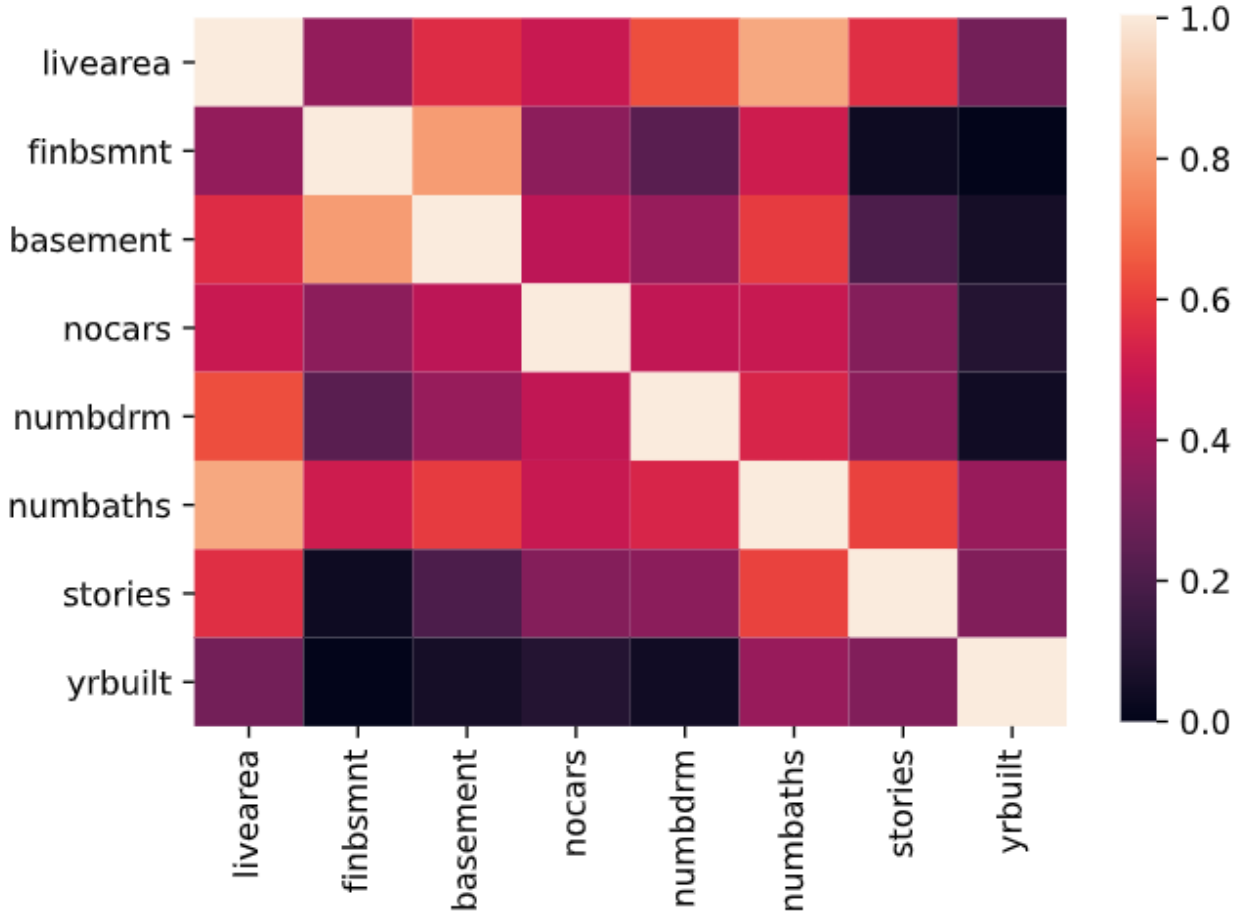
In this project what i demonstrated was an understanding of the fundementals of a machine learning model. What i had to do was start with a raw data set and filter out all irrelevant information that could not be used to predict what year the house was built based off selected variable(s). I used a sample size of 500 with that data to check for corelation and then choose a variable that i could train my model with. With all of this I was able to accuractly predict housing prices to an accuracy of 90 percent.

TECHNICAL DETAILS

GRAND QUESTION 1

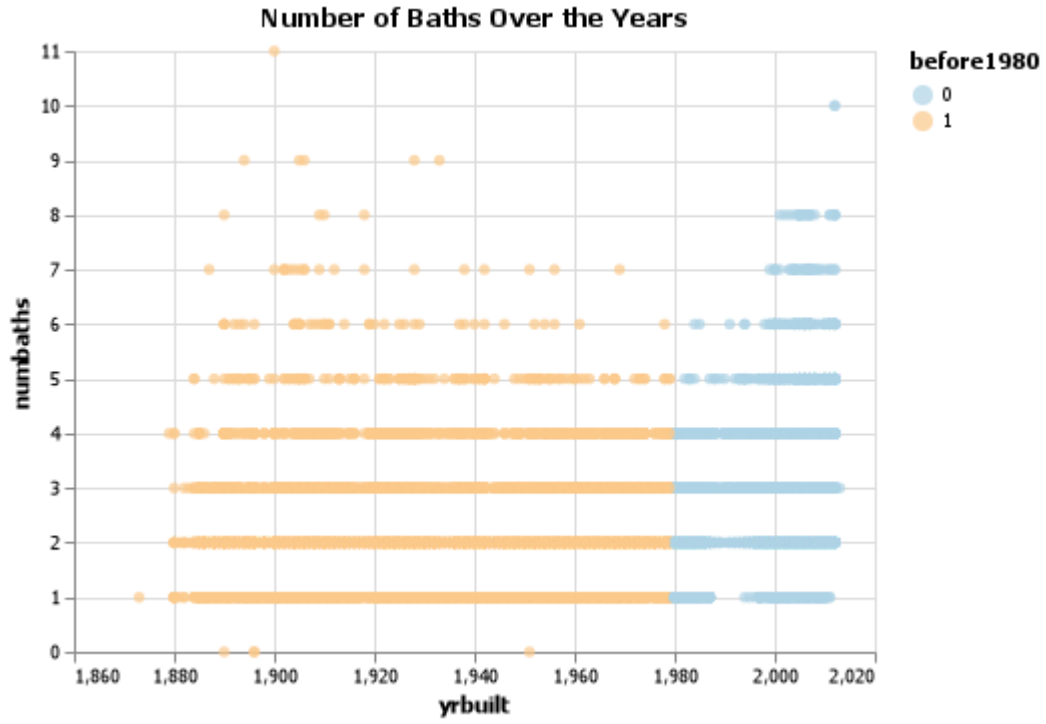
Create 2-3 charts that evaluate potential relationships between the home variables and before1980.

Looking at the potential relationships, we can see from the heat map what are the most closely corelated variables.

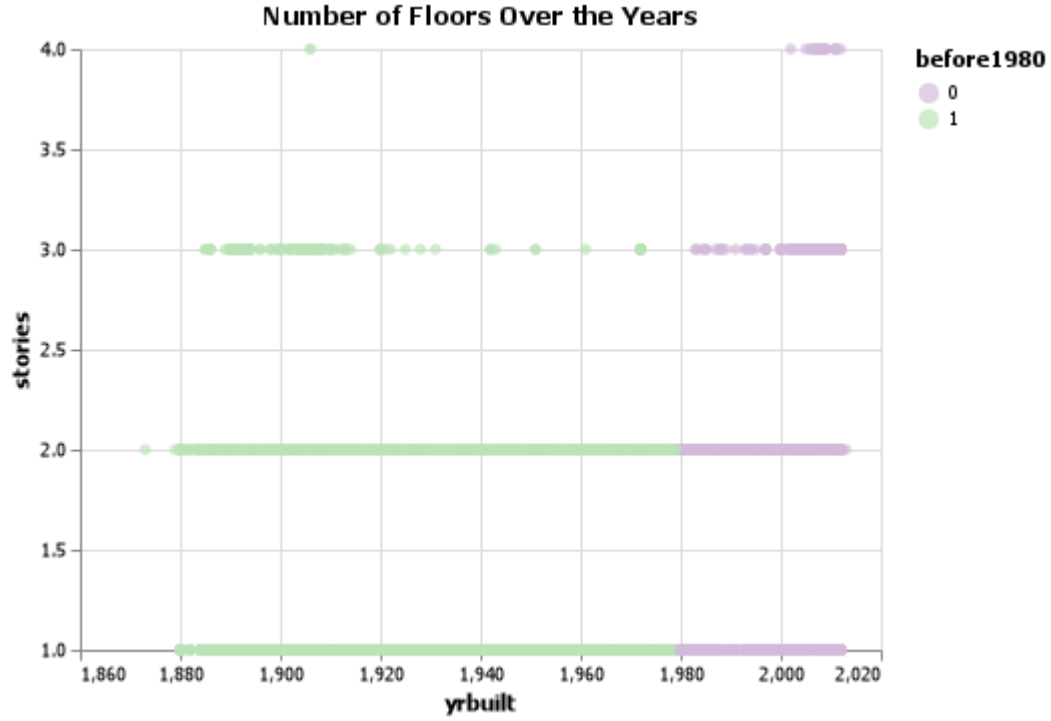


With the lighter colors representing closer corelation, i choose the categories of numbaths and stories to be my two categories i test against. The category of before1980 is a one-hot-encoded variable that will show

wether or not the house was built before 1980. Using this we can graph the correlation.



On the right side of the graph, the 0's or blues represent all the data points past 1980 for the number of baths. On the left side, the 1's or the orange points represent all data points prior to 1980. What we can see from this data is that typically houses built before 1980 had more baths compared to houses built after 1980.



On the right side of the graph, the 0's or purples represent all the data points past 1980 for the number of stories. On the left side, the 1's or the green points represent all data points prior to 1980. What we can conclude from this data is that typically, houses built after 1980 had more stories when compared to houses built prior to 1980.

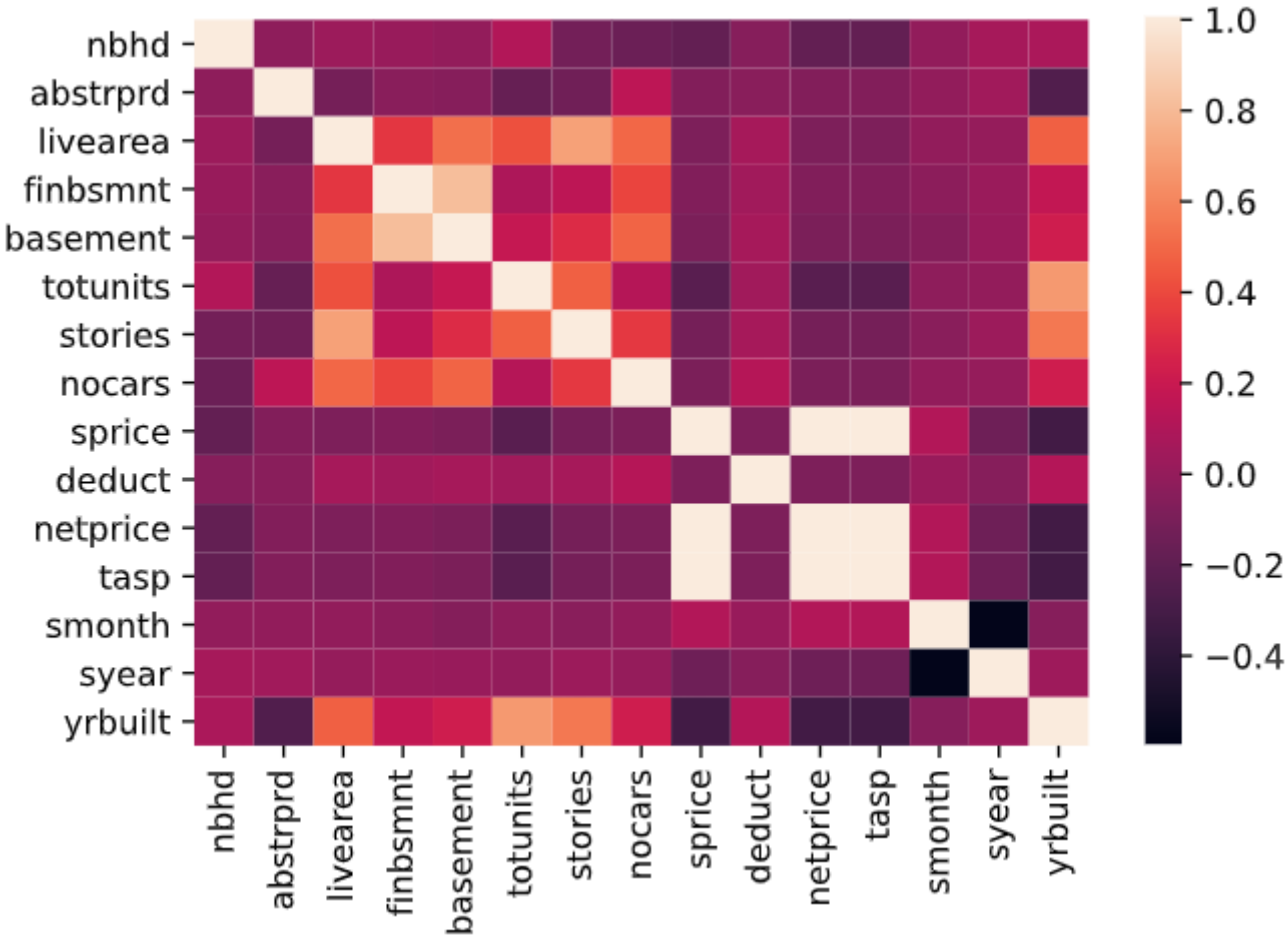
GRAND QUESTION 2

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)?

To start, i sorted out irrelevant data out of the denver dataset. I removed all columns that were classified as objects and also did not include columns with missing data. Then once i filtered out the data to only include relevant data that could be plugged into the model, I added a new column called before1980. I would use this column to train my model to predict wether a house has been built before or after 1980 in Colorado. I did so with the following code.

```
# creates new dataframe with revelant data
colorado_data = dwellings_denver.filter(['nbhd','abstrprd','livearea',
'finbsmnt', 'basement', 'totunits','stories','sprice', 'deduct',
'netprice', 'tasp', 'smonth', 'syear', 'yrbuilt',
'before1980'])
# creates new column in colorado dataframe for before1980
colorado_data['before1980'] = np.where(colorado_data['yrbuilt'] < 1980, 1, 0)
```

I then created a heatmap with all my relevant variables to determine the correlation.



From this data i choose to use the variable tounits to train my model. I choose to choose a confusion matrix to help to illustrate the accuracy of my model. I started by creating my model with the following code.

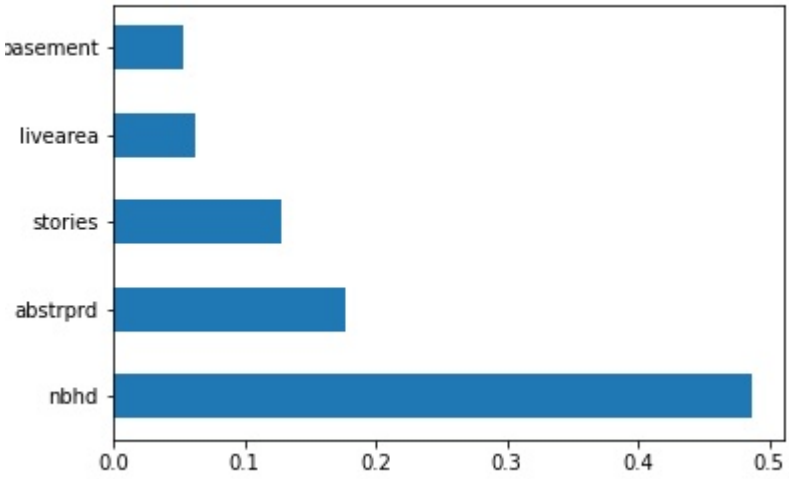
```
X_pred = colorado_data.drop(['yrbuilt', 'before1980'], axis = 1)
y_pred = colorado_data.before1980
X_train, X_test, y_train, y_test = train_test_split(
    X_pred,
    y_pred,
```

```
        test_size = .34,
        random_state = 76)
```

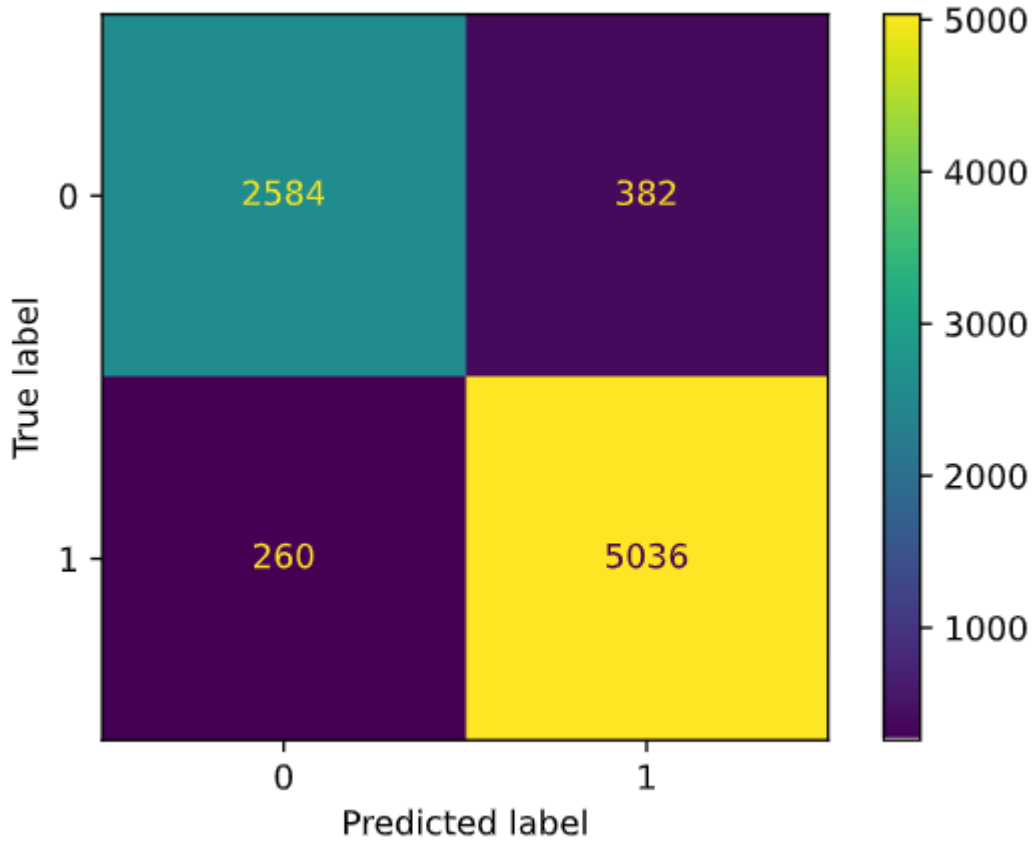
With this model i was able to get a 90 percent accurate model when checking the model against the data.

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 part of a model is the data that is used to train said model. I made sure to have an accurate model by filtering the data that was being fed into the model to assure that I was getting accurate result back. I used a confusion matrix because when comparing between the two categories of before or after 1980, i felt like it was the best way to illustrate if the model was predicting values correctly between the two categories.



T method of confusion matrix makes it easiest to calculate the accuracy of the testing of the model. It is easy to verify the precision, Recall, and f-1 score of our model.

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.

	precision	recall	f1-score	support
0	0.87	0.86	0.86	2884
1	0.92	0.93	0.92	4907
accuracy			0.90	7791
macro avg	0.89	0.89	0.89	7791
weighted avg	0.90	0.90	0.90	7791

	Predicted No	**Predicted Yes**
Actual No	2584	382
Actual Yes	260	5036

What this model also shows us, is that the model is slightly more accurate at predicting houses that are before 1980 compared to houses after 1980.

Precision method

Using this data we can see the precision of the data. We do this by dividing the number of true positives by the sum of the number of true positives and number of false positives (number of true positives / (number of true positive + number of false positives)). Doing this we get a a precision of 0.92 for our model.

Recall method

Using the Recall method, we check by dividing the number of true positives by the sum of the number of true positives and number of false negatives (number of true positive + number of false negatives)). Doing so we get a recall of 0.93.

APPENDIX A (PYTHON SCRIPT)

```
# %%
# the full imports
import pandas as pd
import numpy as np
import seaborn as sns
import altair as alt
import matplotlib.pyplot as plt

# %%
# the from imports
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import GradientBoostingClassifier
from sklearn import metrics

# %%
# imports data
dwellings_denver =
pd.read_csv("https://github.com/byuidatascience/data4dwellings/raw/master/data-
raw/dwellings_denver/dwellings_denver.csv")
dwellings_ml =
pd.read_csv("https://github.com/byuidatascience/data4dwellings/raw/master/data-
raw/dwellings_ml/dwellings_ml.csv")
dwellings_neighborhoods_ml =
pd.read_csv("https://github.com/byuidatascience/data4dwellings/raw/master/data-
raw/dwellings_neighborhoods_ml/dwellings_neighborhoods_ml.csv")
# allows large amounts of json code to be run
alt.data_transformers.enable('json')

# %%
# 1 - Shows corelation of year built to other categories
h_subset = dwellings_ml.filter(['livearea', 'finbsmnt',
    'basement', 'yearbuilt', 'nocars', 'numbdrm', 'numbaths',
    'stories', 'yrbuilt', 'before1980']).sample(500)
# creates chart of all data points
sns.pairplot(h_subset, hue = 'before1980')
corr = h_subset.drop(columns = 'before1980').corr()
```

```
# %%
# 1 - saves correlation heatmap
sns.heatmap(corr).figure.savefig('1980_correlation.jpg')

# %%
# 1 - graph 1 for numbaths to compared to year built
(alt.Chart(dwellings_ml, title = 'Number of Baths Over the Years')
    .encode(
        x = alt.X('yrbuilt', scale = alt.Scale(zero = False)),
        y = alt.Y('numbaths', scale = alt.Scale(zero = False)),
        color = alt.Color('before1980:O' ,scale = alt.Scale(scheme='blueorange')))
        .mark_circle()).save('1980_numbaths_correlation.png')

# %%
# 1 - graph for number of floors compared to year built
(alt.Chart(dwellings_ml, title = 'Number of Floors Over the Years')
    .encode(
        x = alt.X('yrbuilt', scale = alt.Scale(zero = False)),
        y = alt.Y('stories', scale = alt.Scale(zero = False)),
        color = alt.Color('before1980:O' ,scale =
alt.Scale(scheme='purplegreen')))
        .mark_circle()).save('1980_stories_correlation.png')

# %%
# 2 + 3 - creates new dataframe with revelant data
colorado_data = dwellings_denver.filter(['nbhd','abstrprd','livearea',
'finbsmnt', 'basement', 'totunits','stories','sprice', 'deduct',
'netprice', 'tasp', 'smonth', 'syear', 'yrbuilt',
'before1980'])

# %%
# 2 + 3 - creates new column in colorado dataframe for before1980
colorado_data['before1980'] = np.where(colorado_data['yrbuilt'] < 1980, 1, 0)

# %%
# 2 + 3 - creates a corelation chart that shows relationship to year built for
colorado
# gets a sample size from data
colorado_h_subset = colorado_data.sample(500)
# creates correlation chart of all data points
sns.pairplot(colorado_h_subset, hue = 'before1980')
colorado_corr = colorado_h_subset.drop(columns = 'before1980').corr()

# %%
# 2 + 3 - creates and saves heatmap for colorado correlation
sns.heatmap(colorado_corr).figure.savefig('colorado_1980_correlation.jpg')

# %%
# 2 + 3 - creates chart showing the total units over the years in colorado
(alt.Chart(colorado_data, title = 'Denver Home Total Units over the years')
    .encode(
        x = alt.X('yrbuilt', scale = alt.Scale(zero = False)),
        y = alt.Y('totunits', scale = alt.Scale(zero = False)),
        color = alt.Color('before1980:O' ,scale = alt.Scale(scheme='redblue')))
```

```
        .mark_circle()).save('colorado_1980_tstories_correlation.png')

# %%
# 2 + 3 - Sets parameters for model, comparing total units to before1980
X_pred = colorado_data.drop(['yrbuilt', 'before1980'], axis = 1)
y_pred = colorado_data.before1980
X_train, X_test, y_train, y_test = train_test_split(
    X_pred,
    y_pred,
    test_size = .34,
    random_state = 76)

# %%
clf = GradientBoostingClassifier()
clf = clf.fit(X_train, y_train)
predict_p =  clf.predict(X_test)

# %%
# 2 + 3 - creates confusion matrix and prints to markdown
model_confusion_matrix_results = pd.DataFrame(metrics.confusion_matrix(y_test,
predict_p))
print(model_confusion_matrix_results.to_markdown())
metrics.plot_confusion_matrix(clf, X_test, y_test)

# %%
model_results = metrics.classification_report(y_test, predict_p)
print(model_results)

# %%
# 2 + 3 - shows the p values of each column
df_features = (pd.DataFrame(
        {'f_names': X_train.columns,
        'f_values': clf.feature_importances_})
    .sort_values('f_values', ascending = False))
```