CSE 250 Coding Challenge

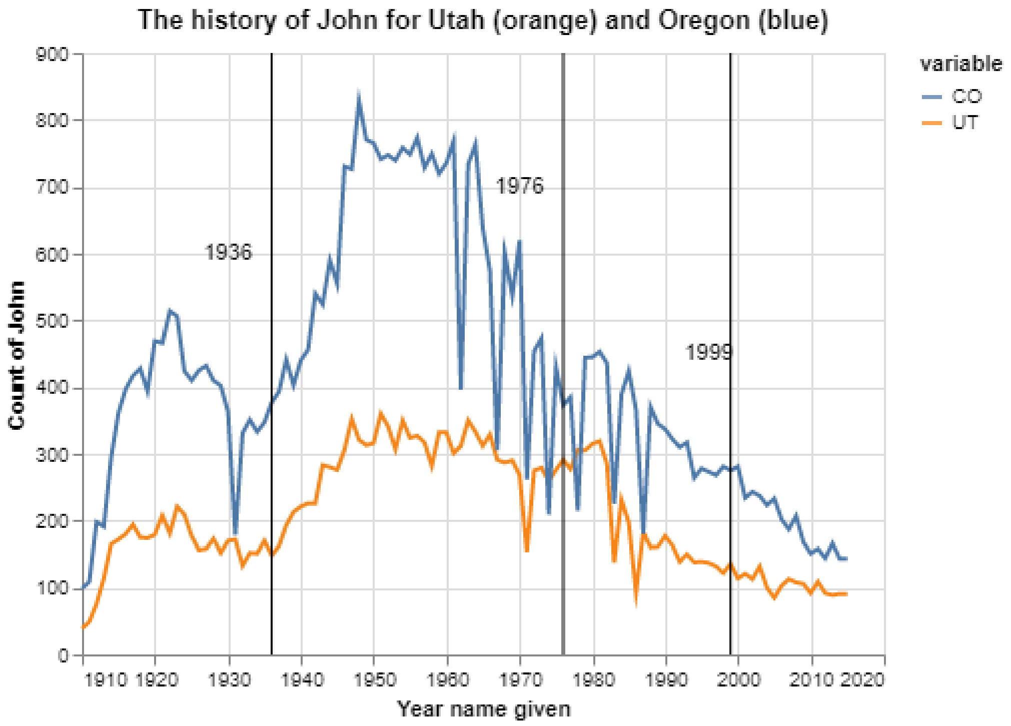
Porter Moody

Challenge Summary

First challenge I filtered down to just the data I needed. Then created a plot with layers.

Challenge 1

Answer



Code

```
#%%
import pandas as pd
import altair as alt
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.ensemble import GradientBoostingClassifier
from sklearn import metrics


#%%

url_names = 'https://github.com/byuidatascience/data4names/raw/master/data-raw/names_year/names_
dat_names = pd.read_csv(url_names)

dat_names

# %%
##### transforming
dat = dat_names
dat = dat.query("name == 'John'").filter(['name', 'year', 'UT', 'CO']).melt(['name','year'])
dat



# %%
c1 = alt.Chart(dat, title = 'The history of John for Utah (orange) and Oregon (blue)').mark_line
    alt.X('year', axis = alt.Axis(format = "d"), title = 'Year name given'),
    alt.Y('value', title = 'Count of John'),
    # color = alt.value('red'),
    color=alt.Color('variable')
)
# .properties(width=700)
c1
#%%

dat_ = pd.DataFrame({
    'x':[1936, 1976, 1999]
})
dat_
#%%
c2 = alt.Chart(dat_).mark_rule().encode(
    alt.X('x'))


(c1 + c2)
### now add year labels
dat_text = pd.DataFrame({
    'text':['1936', '1976', '1999'],
    'year':[1930, 1970, 1996],
    'count' : [600,700,450]
})
text = alt.Chart(dat_text).mark_text().encode(
    alt.X('year'),
    alt.Y('count'),
    text = 'text'
)

chart = (c1 + c2 + text)
chart.save('visuals/first.png')
```

Challenge 2

Answer

####### ####### 2 ...

mean
704.75

...

Code

```
############## 2
mister = pd.Series([np.nan, 15, 22, 45, 31, np.nan, 85, 38, 129, 8000, 21, 2])

## use numpy
median_ = np.nanmedian(mister)
median_
clean = mister.fillna(median_)
final_mean = np.mean(clean)
print(pd.DataFrame({final_mean}).to_markdown())
```

Challenge 3

Answer

Code

```
# dwellings_ml.groupby(['stories']).agg()
dat = (dwellings_ml.filter([ 'stories', 'nocars']).query("nocars <= 4")
            .groupby(['stories','nocars']).agg(['count'])).reset_index()

dat = dat.pivot(values='stories', columns='nocars')
dat
```

Challenge 4

Answer

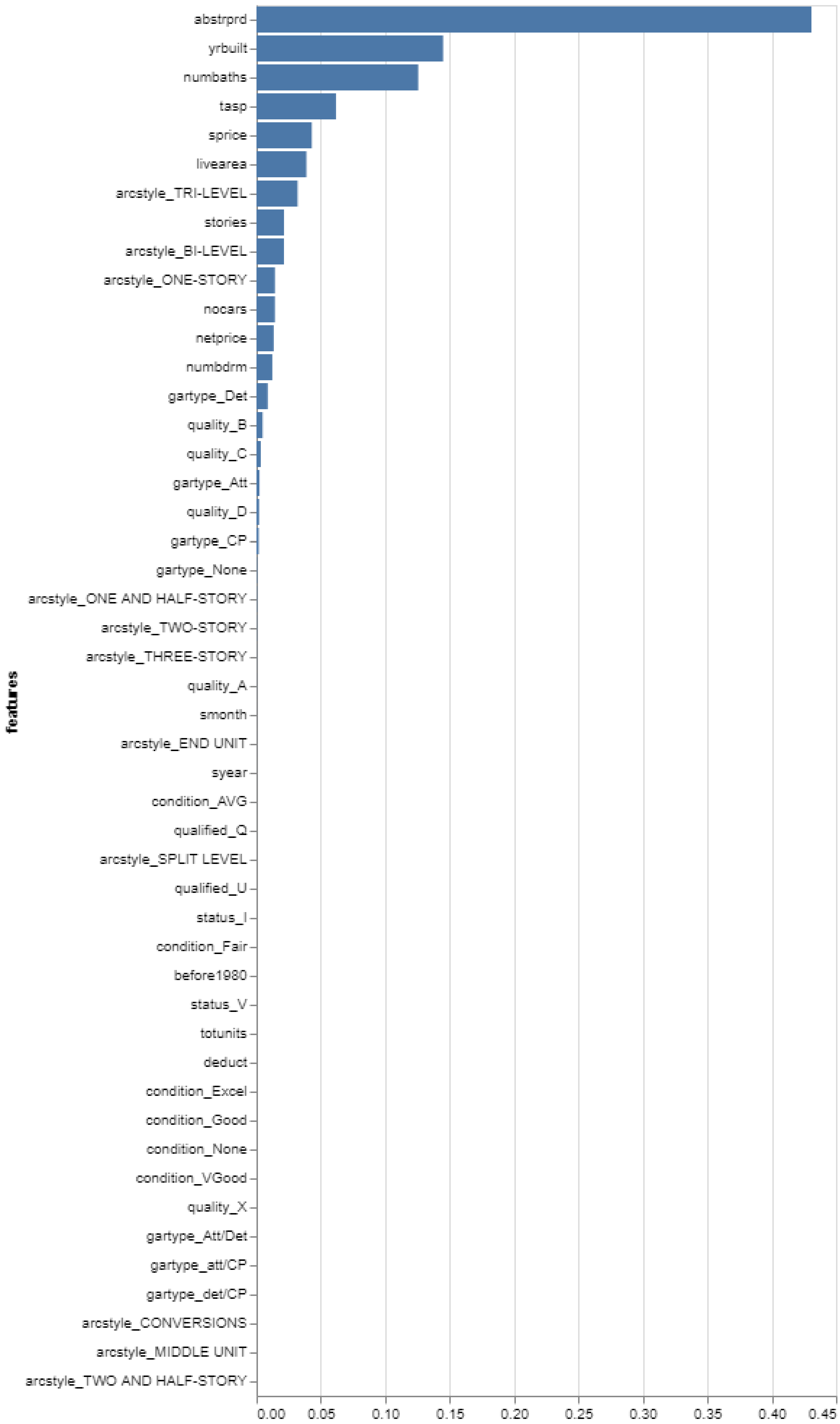
standard deviation
15.2201

```
####### 4
mother = pd.Series(['N/A', 15, 22, 45, 31, -999, 21, 2, 0, 0, 0, 'broken'])


fixed = mother.replace('N/A' ,np.nan).replace('broken',np.nan).replace(-999, np.nan)
np.std(fixed)
```

Challenge 5

Answer


features
abstrprd
yrbuilt
numbaths
tasp
sprice
livearea
arcstyle_TRI-LEVEL
stories
arcstyle_BI-LEVEL
arcstyle_ONE-STORY
nocars
netprice
numbdrm
gartype_Det
quality_B
quality_C
gartype_Att
quality_D
gartype_CP
gartype_None
arcstyle_ONE AND HALF-STORY
arcstyle_TWO-STORY
arcstyle_THREE-STORY
quality_A
smonth
arcstyle_END UNIT
syear
condition_AVG
qualified_Q
arcstyle_SPLIT LEVEL
qualified_U
status_I
condition_Fair
before1980
status_V
totunits
deduct
condition_Excel
condition_Good
condition_None
condition_VGood
quality_X
gartype_Att/Det
gartype_att/CP
gartype_det/CP
arcstyle_CONVERSIONS
arcstyle_MIDDLE UNIT
arcstyle_TWO AND HALF-STORY
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45

```
dwellings_ml = pd.read_csv("https://github.com/byuidatascience/data4dwellings/raw/master/data-ra

X = dwellings_ml.drop(dwellings_ml.filter(regex = 'basement|finbsmnt|BASEMENT').columns, axis =
y = dwellings_ml.basement
y[y > 0] = 1
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size = .2, random_state = 76)

boost = GradientBoostingClassifier(random_state = 76)


boost.fit(X_train, y_train)
y_pred_boost = boost.predict(X_test)

dat_features_boost = pd.DataFrame({
    "values" : boost.feature_importances_,
    "features" : X_train.columns
})

rank_boost = (alt.Chart(dat_features_boost, title="")
    .encode(
        alt.X('values'),
        alt.Y('features', sort = "-x"))
    .mark_bar()
)
rank_boost
```