KAILI HAMILTON MARCH 23, 2023

CAPSTONE 2 | MACHINE LEARNING

MACHINE LEARNING CYCLE

1. **Gathering Data**
    a. Question 1: Given certain variables, can I predict which brand of shoe (off-white vs yeezy)?
        i. Classification problem
        ii. Not a useful question in any context
        iii. Brand influences the sale price (or resale price), not the other way around.
        iv. Try a different question.
    b. Question 2: Can I predict when to resale a sneaker in order to maximize the profit (sale price – retail price)?
        i. Tried dropping a bunch of columns to see what would happen to the r^2 value (which is how much of the variance our model can explain)
        ii. Tried to interpret the coefficients in context but it didn't make sense.
        iii. Realized that this question may be beyond the scope of linear regression. Likely requires forecasting and time series.
        iv. Try a different question.
    c. Question 3: What variables contribute most to the profit margin for each sneaker brand, Off-White or Yeezy.

2. **Data Exploration**
    a. I had explored the data for the Capstone 1 course, so I was familiar with the data set.
    b. I cleaned the data as well, but I need to do some feature engineering to use for ML.

3. **Pre-process Data**
    a. Changed all data types to numerical values and dropped irrelevant strings.
        i. I used LabelEncoder to turn Brand into 0 and 1.
        ii. Used get_dummies for Silhouette and concatenated that into the dataframe.
        iii. Added a column Price Ratio (Sale Price / Retail Price)

4. **Building Machine Learning Models**

a. Ordinary Least Squares Regression Split data into training (80%) and testing sets (20%)
b. Supervised learning

5. **Train the Model**
   a. Only know how to do OLS
   b. Perhaps the data isn't linear or should be doing a different type of linear regression.

6. **Evaluate the Model**
   a. Look at r^2 value, f-statistic, p-value, and confidence interval to determine if model is a good fit.
   b. Added and deleted various columns to see if it improves the model.
   c. Went back and created a 'Price Ratio' column (Sale Price / Retail Price) which improved the model over 'Profit Margin'
   d. Best fit + lowest MSE
      i. Brand, Shoe Size, Constant vs Price Ratio
   e. Filtered by Brand – Off-White Nikes had a better fit than Yeezy's

7. **Make Predictions**
   a. Used the test data to predict
   b. MSE for predicted price ratio is 1.37 so square root of that is 1.17
      i. This is an improvement on the 1.5 MSE when the constant wasn't included
   c. Repeated the process for one brand at a time

8. **Conclusions**
   a. Had to change my original question to fit what I could do with the day. Added a price ratio column:
      i. Price ratio is sale price / retail price. This is basically a percentage of the retail price that the shoe was sold for. A positive means sold above retail, a negative means sold below retail.
   b. The r^2 value when including shoe size, constant term, and Brand is 0.426. This means that about 42.6% of the variance can be explained with my model. The r-value is 0.65, which is indicating a moderate, positive correlation between the explanatory and response variables.
   c. The MSE for the above variables is the lowest, 1.37, the square root of that being 1.17. This means that on average, the predicted values have a price ratio error of 1.17, i.e., on average predicted values are 117% above or below the actual sale price.
   d. The r^2 and the MSE are the best fit when running different columns, but still not a great fit. The error is too high. If a shoe retails for $100, and actually sales for $200 (price ratio of 2), then my model would, on

average, have an error for price ratio of ± 1.17 above or below that actual value of 2. So price ratio of 3.17 or 0.83. It could have predicted the sale price to be $317 or $83. This is a big interval for sneaker prices.

e. Can't confidently say I can predict the price ratio given various factors, such as the brand and shoe size. But, there where some interesting and useful insights.
f. But, what I can see is that the brand of the shoe is the most influential factor in increasing the price ratio
    i. The coefficient is 2.177 for the brand. This means that for the sneaker designated as a 1 (Off-White Nike), will increase the price ratio of 217%. So, the Off-White Nikes are generally more profitable than the Yeezy's
g. I filtered the data set by brand to see if there was any changes. Yeezy's had r^2 was 0.18 but MSE was 0.36. Off-White had a higher r^2 of 0.58 and a higher MSE 1.5.
    i. V1 silhouette is more influential in increasing the price ratio than V2's for Yeezys (2.9 vs -0.7)
    ii. Nike Air Force 1 Low Virgil Abloh most influential silhouette with 3.1 price ratio increase on average.

[144]:
```python
#X = df[['Brand', 'Shoe Size', 'Elapsed Time Days', 'Sale Price', 'Retail Price']]
X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Price Ratio']

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=2)
```

[145]:
```python
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[145]:

OLS Regression Results

Dep. Variable:	Price Ratio	**R-squared:**	0.994
Model:	OLS	**Adj. R-squared:**	0.994
Method:	Least Squares	**F-statistic:**	8.567e+05
Date:	Tue, 21 Mar 2023	**Prob (F-statistic):**	0.00
Time:	17:58:01	**Log-Likelihood:**	57307.
No. Observations:	79964	**AIC:**	-1.146e+05
Df Residuals:	79948	**BIC:**	-1.144e+05
Df Model:	15		
Covariance Type:	nonrobust		

[146]:
```python
X = df[['Brand', 'Shoe Size', 'Elapsed Time Days', 'Sale Price', 'Retail Price']]
#X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Price Ratio']

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=2)
```

[147]:
```python
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[147]:

OLS Regression Results

Dep. Variable:	Price Ratio	**R-squared (uncentered):**	0.985
Model:	OLS	**Adj. R-squared (uncentered):**	0.985
Method:	Least Squares	**F-statistic:**	1.086e+06
Date:	Tue, 21 Mar 2023	**Prob (F-statistic):**	0.00
Time:	18:11:30	**Log-Likelihood:**	-23784.
No. Observations:	79964	**AIC:**	4.758e+04
Df Residuals:	79959	**BIC:**	4.762e+04
Df Model:	5		
Covariance Type:	nonrobust		

[148]:

```
X = df[['Brand', 'Shoe Size', 'Elapsed Time Days']]
#X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Price Ratio']

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=2)
```

[149]:

```
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[149]:

OLS Regression Results

Dep. Variable:	Price Ratio	**R-squared (uncentered):**	0.812
Model:	OLS	**Adj. R-squared (uncentered):**	0.812
Method:	Least Squares	**F-statistic:**	1.149e+05
Date:	Tue, 21 Mar 2023	**Prob (F-statistic):**	0.00
Time:	18:12:22	**Log-Likelihood:**	-1.2628e+05
No. Observations:	79964	**AIC:**	2.526e+05
Df Residuals:	79961	**BIC:**	2.526e+05
Df Model:	3		
Covariance Type:	nonrobust		

[150]:

```
X = df[['Brand', 'Shoe Size']]
#X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Price Ratio']

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=2)
```

[151]:

```
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[151]:

OLS Regression Results

Dep. Variable:	Price Ratio	**R-squared (uncentered):**	0.807
Model:	OLS	**Adj. R-squared (uncentered):**	0.807
Method:	Least Squares	**F-statistic:**	1.667e+05
Date:	Tue, 21 Mar 2023	**Prob (F-statistic):**	0.00
Time:	18:13:00	**Log-Likelihood:**	-1.2735e+05
No. Observations:	79964	**AIC:**	2.547e+05
Df Residuals:	79962	**BIC:**	2.547e+05
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Brand	2.2396	0.009	238.842	0.000	2.221	2.258
Shoe Size	0.1649	0.001	320.777	0.000	0.164	0.166

[200]:
```python
X = df[['Brand', 'Shoe Size', 'const']]
#X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Price Ratio']

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=2)
```

[201]:
```python
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[201]:

OLS Regression Results

Dep. Variable:	Price Ratio	**R-squared:**	0.426
Model:	OLS	**Adj. R-squared:**	0.426
Method:	Least Squares	**F-statistic:**	2.972e+04
Date:	Thu, 23 Mar 2023	**Prob (F-statistic):**	0.00
Time:	06:56:52	**Log-Likelihood:**	-1.2382e+05
No. Observations:	79964	**AIC:**	2.476e+05
Df Residuals:	79961	**BIC:**	2.477e+05
Df Model:	2		
Covariance Type:	nonrobust		

[157]:

```
X = df[['Brand', 'Shoe Size']]
#X = df.drop(['Price Ratio', 'const'], axis=1)
y = df['Profit Margin']

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=2)
```

[158]:

```
lr = sm.OLS(y_train, X_train).fit()
lr.summary()

#looks like the datetime values didn't work. Tried dropping
```

[158]:

OLS Regression Results

Dep. Variable:	Profit Margin	**R-squared (uncentered):**	0.639
Model:	OLS	**Adj. R-squared (uncentered):**	0.639
Method:	Least Squares	**F-statistic:**	7.062e+04
Date:	Tue, 21 Mar 2023	**Prob (F-statistic):**	0.00
Time:	18:16:43	**Log-Likelihood:**	-5.4264e+05
No. Observations:	79964	**AIC:**	1.085e+06
Df Residuals:	79962	**BIC:**	1.085e+06
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Brand	353.9026	1.689	209.592	0.000	350.593	357.212
Shoe Size	14.3112	0.093	154.576	0.000	14.130	14.493