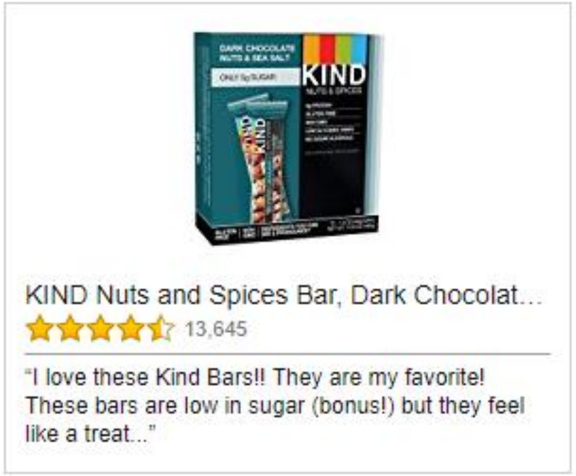
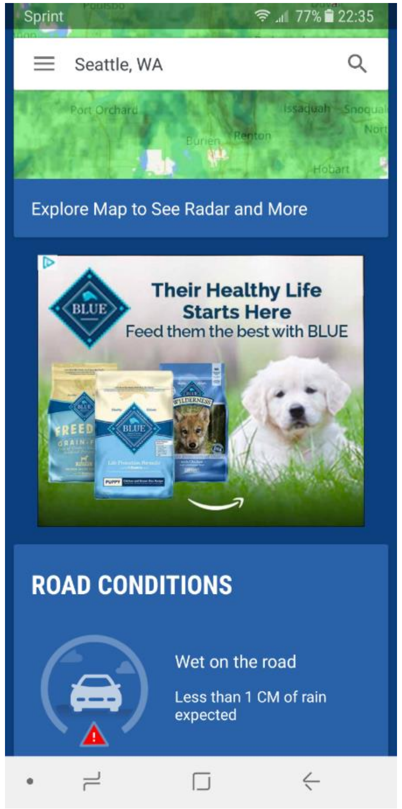
Ad Campaign Performance Predictive Model

Jessica Otaguro

Amazon Advertising Platform

- Allows advertisers to reach Amazon customers across the web through display and video ads





Can we predict return on ad spend based on campaign and product attributes such as campaign budget, time of year, product rating, and price?

Data

- Wrote script to scrape product category, price, rating, and number of reviews from Amazon.com for 12k ASINs

```
asins = pd.read_csv('asins.csv')
asindetails = pd.DataFrame(columns=['asin','rating'])
driver = webdriver.Chrome()
driver.implicitly_wait(10)
for index, row in asins.iterrows():
    if index >= 12430:
        baseurl = 'https://www.amazon.com/exec/obidos/ASIN/'
        driver.get(baseurl + str(row[0]))
        try:
            WebDriverWait(driver, 10).until(
                EC.presence_of_element_located((By.XPATH, "//*[@id='reviewSummary']/div[2]/span"))
                )
        except:
            continue
        temp = pd.DataFrame({
            'asin': row[0],\
            'rating': [driver.find_element_by_xpath("//*[@id='reviewSummary']/div[2]/span").text.replace(' out of 5 stars','')]
            })
        asindetails = asindetails.append(temp)
        asindetails.to_csv("asinrating12430.csv")
print ("Job Complete")
```

Data

Field	Data Type	Description	Example
advertiser_name	Object (string)	Advertiser running the campaign	Toyota
ad_campaign_id	Numeric (integer)	Campaign unique identifier	335234235
startmonth	Numeric (integer)	Month of campaign start date	2
endmonth	Numeric (integer)	Month of campaign end date	5
campaignlength	Numeric (integer)	Length of campaign (End date – start date in days)	31
retargeting	Numeric (integer)	Yes/no if a retargeting line exists in the campaign	1
category	Object (string)	Highest level product category	Electronics
price	Numeric (float)	Average price of campaign ASINs	19.95
reviews	Numeric (float)	Average number of reviews	386.4
rating	Numeric (float)	Average rating (1-5 possible)	4.3
spend	Numeric (float)	Total ad spend	21442.12
sales	Numeric (float)	Total retail sales attributed to campaign	44002.06
dailyspend	Numeric (float)	Ad spend per day (spend / campaign length)	596.43
roas	Numeric (float)	Return on ad spend (sales / spend)	$9.64

```
campaigns = pd.read_csv('campaigns.csv', index_col=0)
campaigns['campaign_start'] = pd.to_datetime(campaigns['campaign_start'])
campaigns['campaign_end'] = pd.to_datetime(campaigns['campaign_end'])
campaigns['roas'] = pd.to_numeric(campaigns['roas'])
campaigns['startmonth'] = campaigns.campaign_start.dt.month
campaigns['endmonth'] = campaigns.campaign_end.dt.month
campaigns['campaignlength'] = (campaigns.campaign_end - campaigns.campaign_start)
campaigns['campaignlength'] = campaigns.campaignlength.dt.days
campaigns = campaigns[(campaigns.campaignlength >= 1 )]
campaigns['dailyspend'] = (campaigns.spend / campaigns.campaignlength)
campaigns.rating.fillna(campaigns.rating.median(), inplace=True)
campaigns.reviews.fillna(campaigns.reviews.median(), inplace=True)
campaigns.price.fillna(campaigns.price.median(), inplace=True)
campaigns = campaigns.dropna(axis=0, how='any')
campaigns = campaigns.drop(['campaign_start','campaign_end'], axis=1)
```

```
advertiser_dummies = pd.get_dummies(campaigns.advertiser_name, prefix='advertiser')
cat_dummies = pd.get_dummies(campaigns.category, prefix='category')
completecampaigns = pd.concat([campaigns, cat_dummies], axis=1)
completecampaigns = pd.concat([completecampaigns, advertiser_dummies], axis=1)
```

```
X = completecampaigns.drop(['advertiser_name','ad_campaign_id','spend','sales','roas','category'], axis=1)
y = completecampaigns.roas
```

Modeling

1. Tested linear regression, k nearest neighbor, decision trees, and random forest, bagging estimators
2. Used all features, features w/o dummies, important features

Linear regression

```
# Important features without dummies
X_important = X[['dailyspend','rating','endmonth']]
scores = cross_val_score(linreg, X_important, y, cv=10, scoring='neg_mean_squared_error')
np.mean(np.sqrt(-scores))
```

8.2410628082427735

```
# Important features with dummies
X_important_dum = X_dummies[['dailyspend','rating','reviews','campaignlength','price']]
scores = cross_val_score(linreg, X_important_dum, y_dummies, cv=10, scoring='neg_mean_squared_error')
np.mean(np.sqrt(-scores))
```

8.4071000557490354

Random forest

```
# All features with dummies
rfreg = RandomForestRegressor(n_estimators=10, max_features=1, oob_score=True)
scores = cross_val_score(rfreg, X_dummies, y_dummies, cv=10, scoring='neg_mean_squared_error')
np.mean(np.sqrt(-scores))
```

8.0319549336602822

```
# Important features without dummies
X_important = X[['reviews','dailyspend','rating','price','campaignlength','endmonth']]
rfreg = RandomForestRegressor(n_estimators=50, max_features=1, oob_score=True)
scores = cross_val_score(rfreg, X_important, y, cv=10, scoring='neg_mean_squared_error')
np.mean(np.sqrt(-scores))
```

8.0273066834033955

KNN

```
# Non-dummies are better than dummies
knn = KNeighborsRegressor(n_neighbors=335)
scores = cross_val_score(knn, X, y, cv=10, scoring='neg_mean_squared_error')
np.mean(np.sqrt(-scores))
```

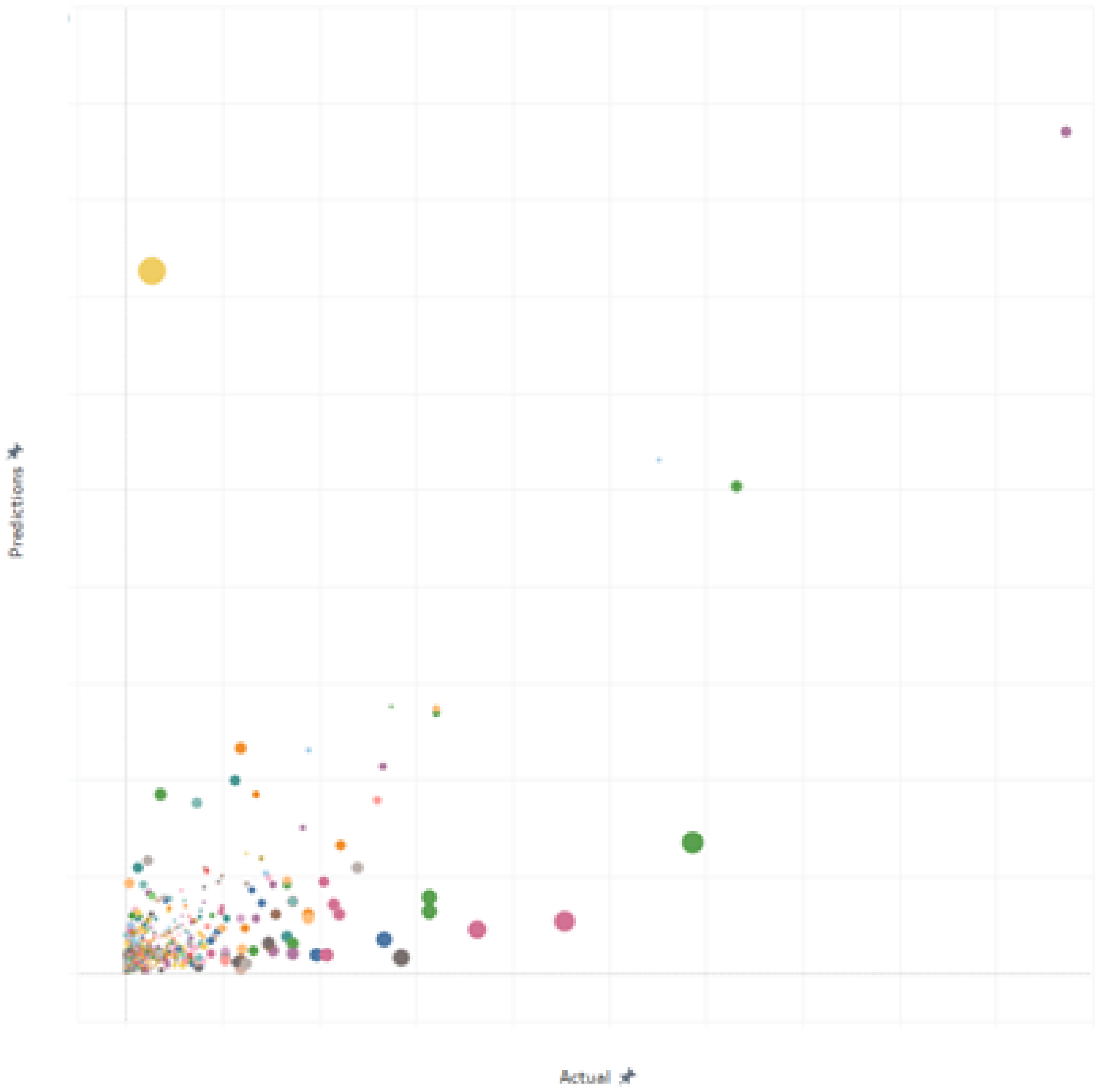
8.1459852516879945

Modeling

	Features incl. dummies		Features not incl. dummies	
	All	Important	All	Important
LinearRegression	128,447,239.61	8.41	8.44	8.24
RandomForestRegressor	7.67	8.21	7.94	7.92
KNeigborsRegressor	8.15	8.15	8.15	8.15
BaggingRegressor(DecisionTreeRegressor)	4.88	6.65	6.61	6.61
BaggingRegressor(KNeigborsRegressor)	7.29	7.29	7.36	7.37
BaggingRegressor(RandomForestRegressor)	5.29	6.72	6.71	6.72
BaggingRegressor(LinearRegression)	25,581,479.72	7.44	7.67	7.44

Modeling

Out of bag error = 0.35
Mean Absolute Error = 2.01
Root Mean Squared Error = 4.88
Null RMSE = 6.65



Next Steps

The model was able to predict return on ad spend fairly accurately for our testing set of campaigns, but can be improved and can evolve into a more accurate, scalable, and extensive model with the below next steps.

- Remove outliers from data set, hone features, and try additional estimators to make model more accurate
- Build distinct models for additional KPIs, regions, and entities
- Test model accuracy on future campaigns
- Incorporate additional features such as supply sources, segments, bids, etc.
- Capture only hero ASIN instead of all tracked ASINs
- Ingest retail product data systematically
- Build a tool on top of the script so that users can input campaign details and receive a prediction