DRIVING GROWTH

COMPANY A CASE

Optimizing Long Term Profit

- Contents
  - Optimizing CAC with targeted ads and customer segmentation
  - Optimizing LTV with personalized offer powered by recommender system
  - Deployment demo
  - Methodology

OPTIMIZING CAC

TARGETED ADS

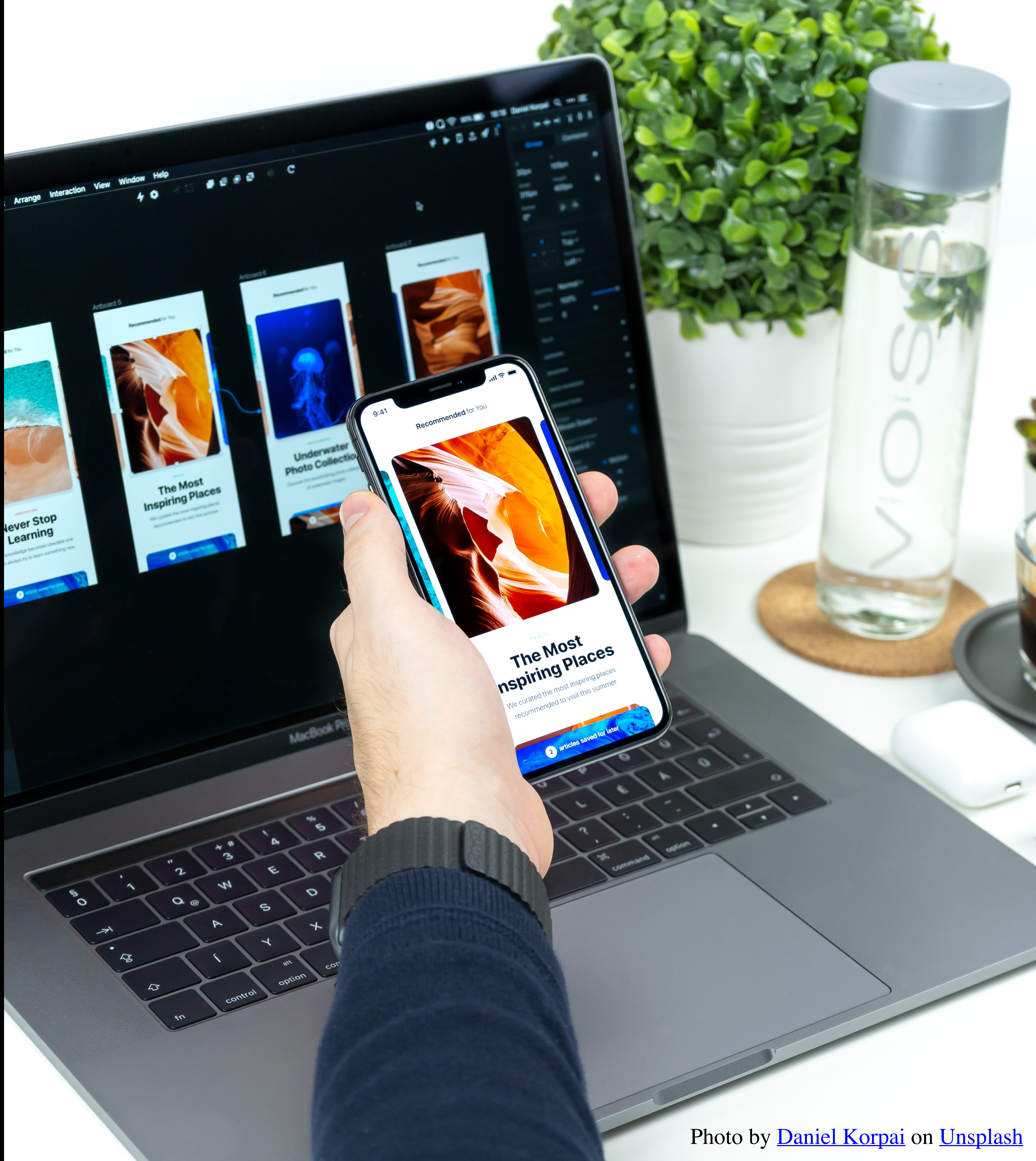


Photo by Daniel Korpai on Unsplash

Google facebook YAHOO! Microsoft

MediaMath datalogix turn RUBICON PubMatic Make every impression count

OpenX pulsepoint m6d appnexus Jumptap Admeld

INDEX lijit AddThis tubemogul More Play Time EVIDON .FOX NETWORKS

EXPERIMENT DESIGN

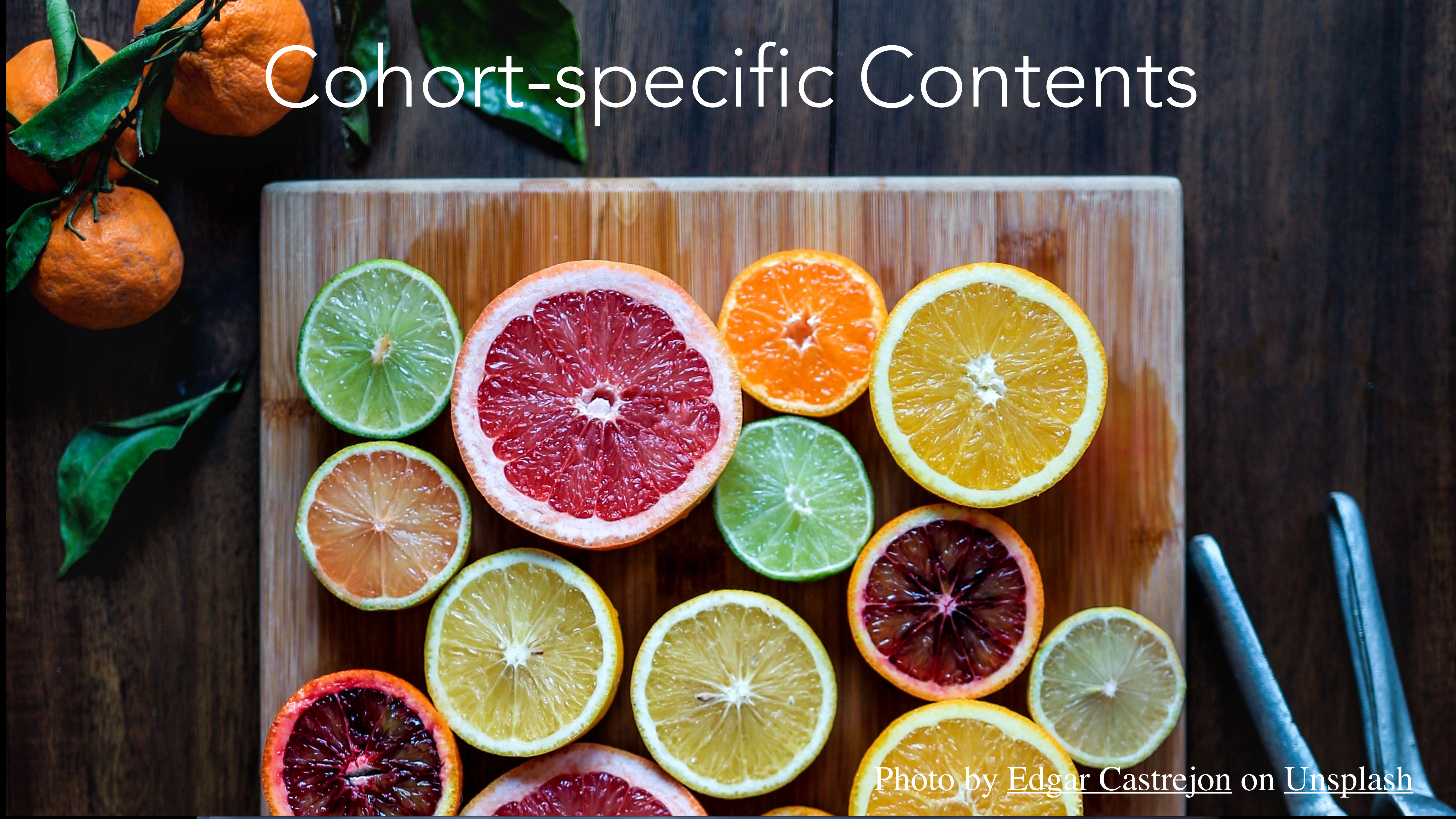
A/B testing of cohort-specific contents vs. top selling product content

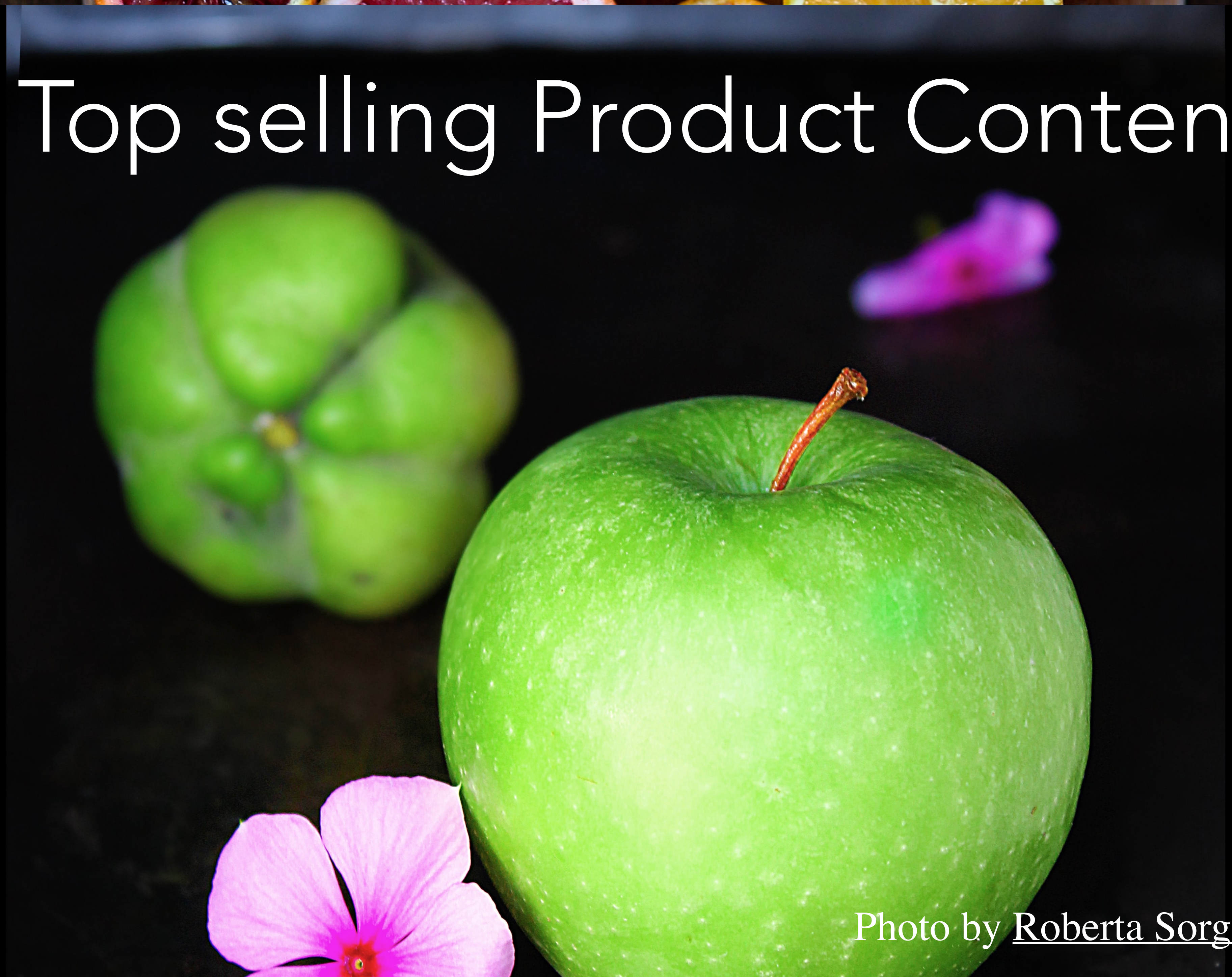
MEASUREMENTS

Ad-click landing page per dollar spent
Page scroll 70% per dollar spent
Product document download per dollar spent

Level of engagement can be converted into implied new customer gained by looking at historical data e.g. 2% of Page scroller ended up buying a policy.

CAC = #new customers / $ ads spent





CONTENTS BY CUSTOMER COHORT

('c.25-29', 'm', 'region_3'):

{'prod_rec': 'savings',

'median_insured_amt': 500,000},

('c.25-29', 'm', 'region_8'):

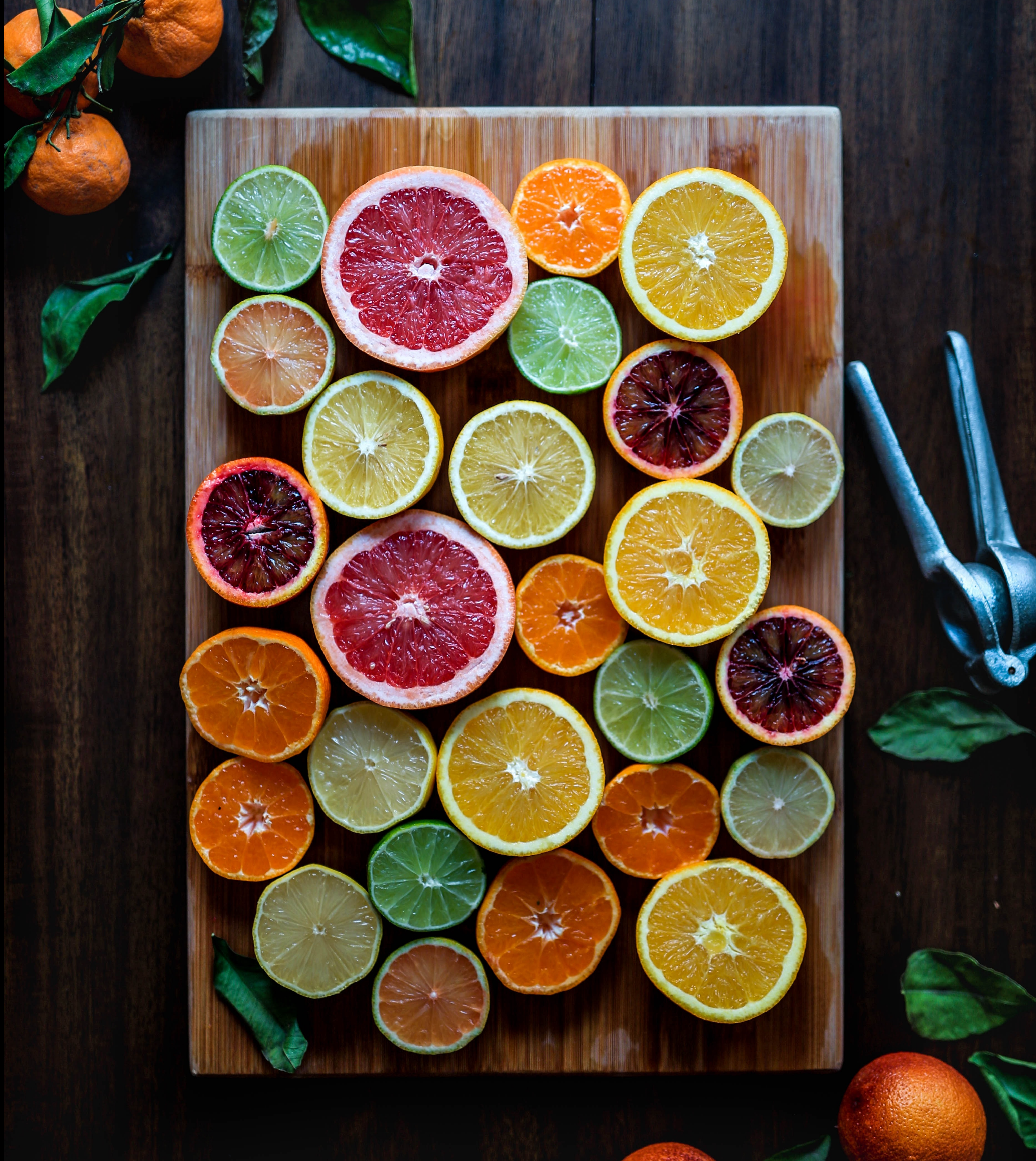
{'prod_rec': 'health',

'median_insured_amt': 25,000},

('d.30-34', 'f', ‘region_1'):

{'prod_rec': 'retirement',

'median_insured_amt': 150,000}



OPTIMIZING LTV

PERSONALIZED XSELL OFFER

POWERED BY RECOMMENDER SYSTEM

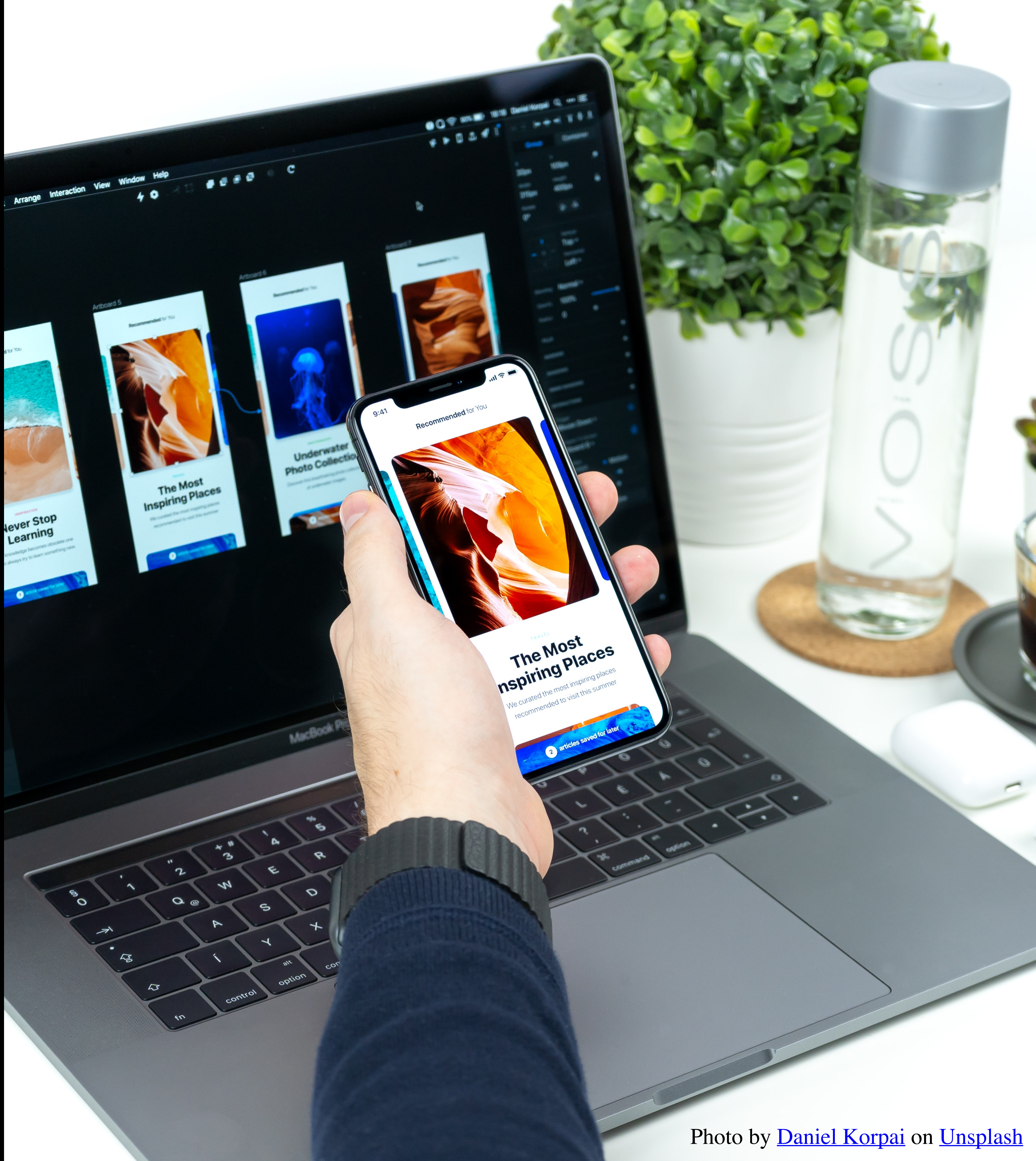


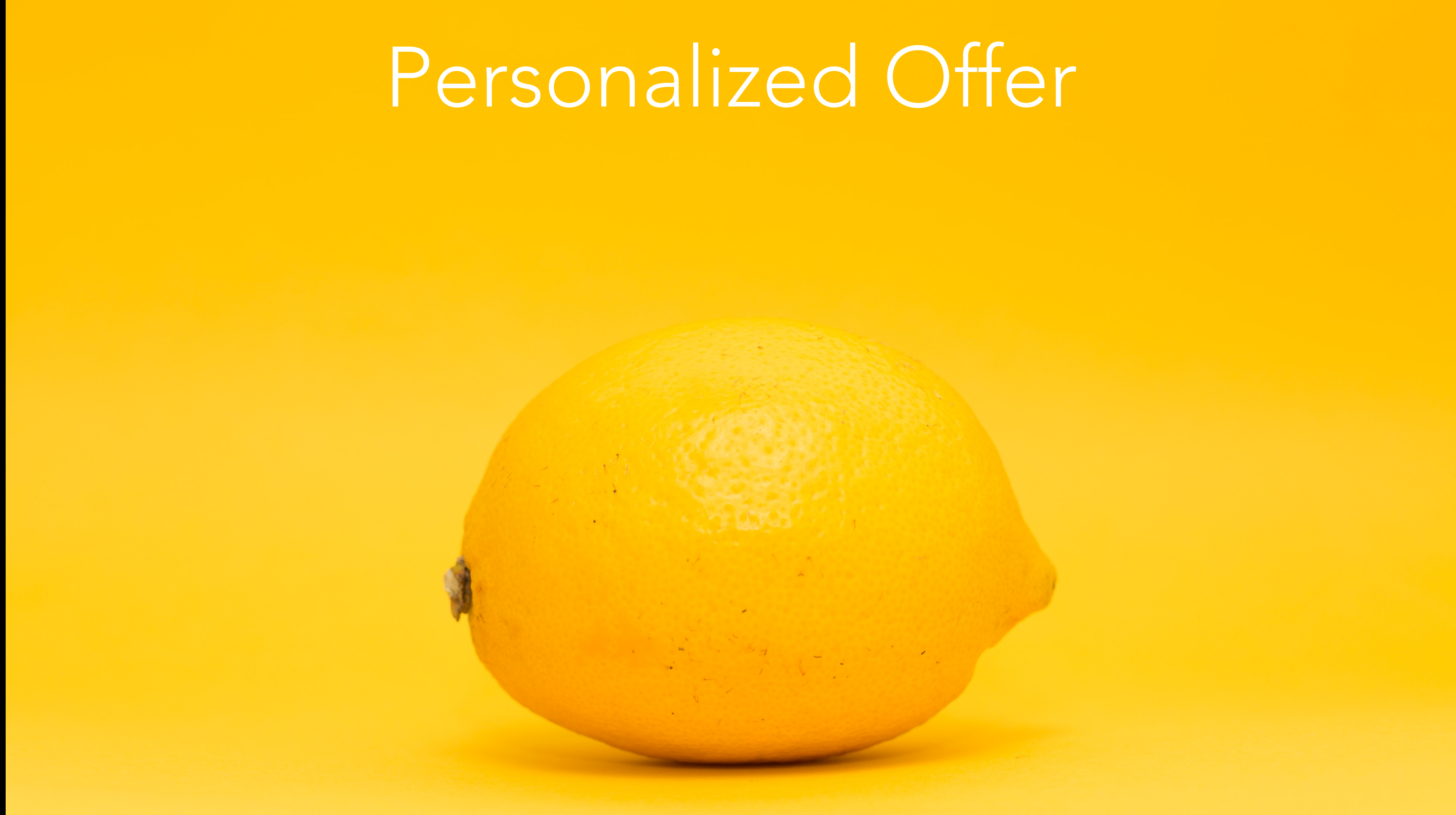
Photo by Daniel Korpai on Unsplash

EXPERIMENT DESIGN

A/B testing of offering product recommended by the model vs. top selling product content

MEASUREMENTS

- uplift in conversion rate (#policy sold/ #offer)
- quarterly measure of LTV per customer
  (present value of expected premium paid throughout customer's expected tenure less present value of expected claims)



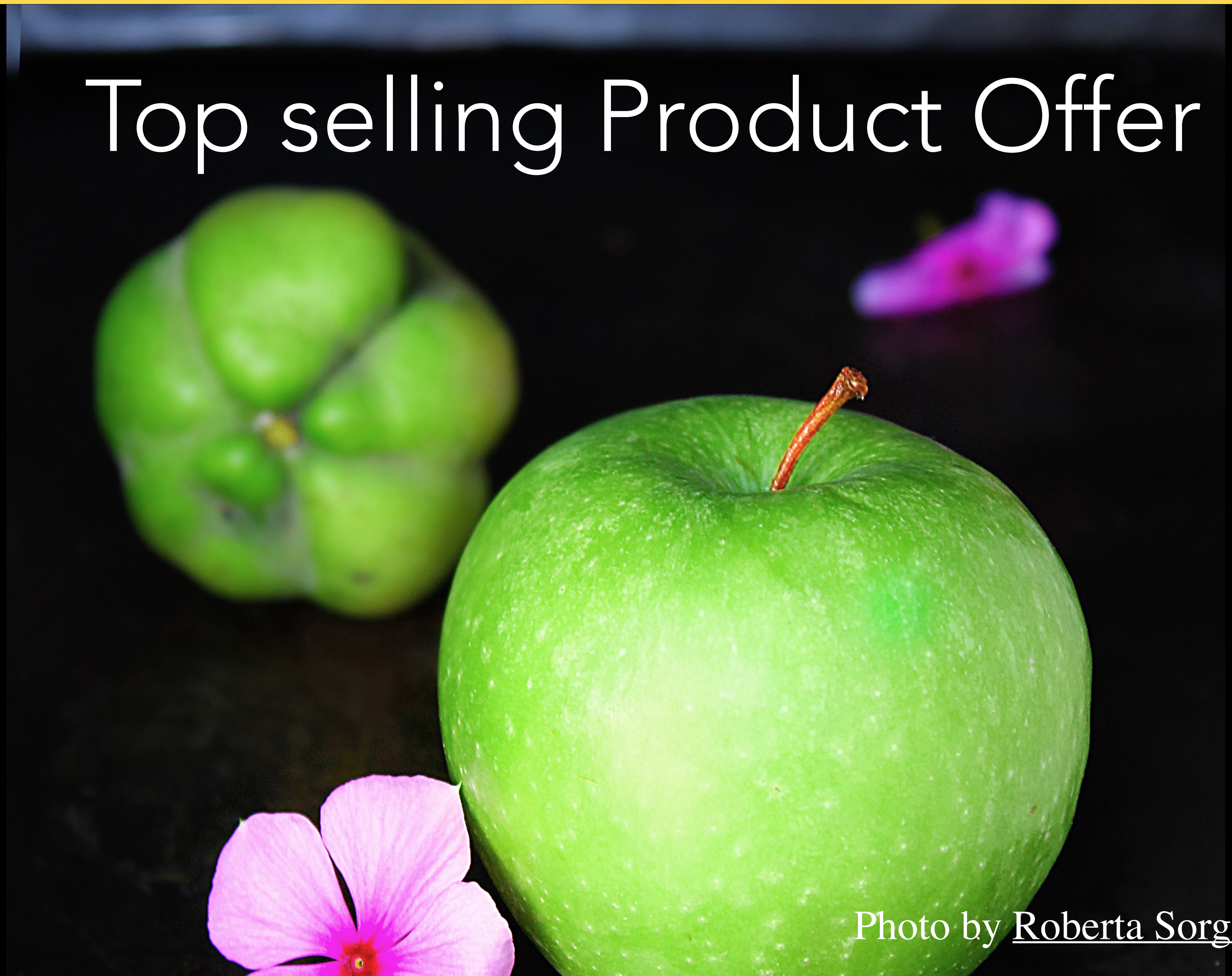


Photo by Roberta Sorge on Unsplash

DEPLOYMENT DEMO

METHODOLOGY

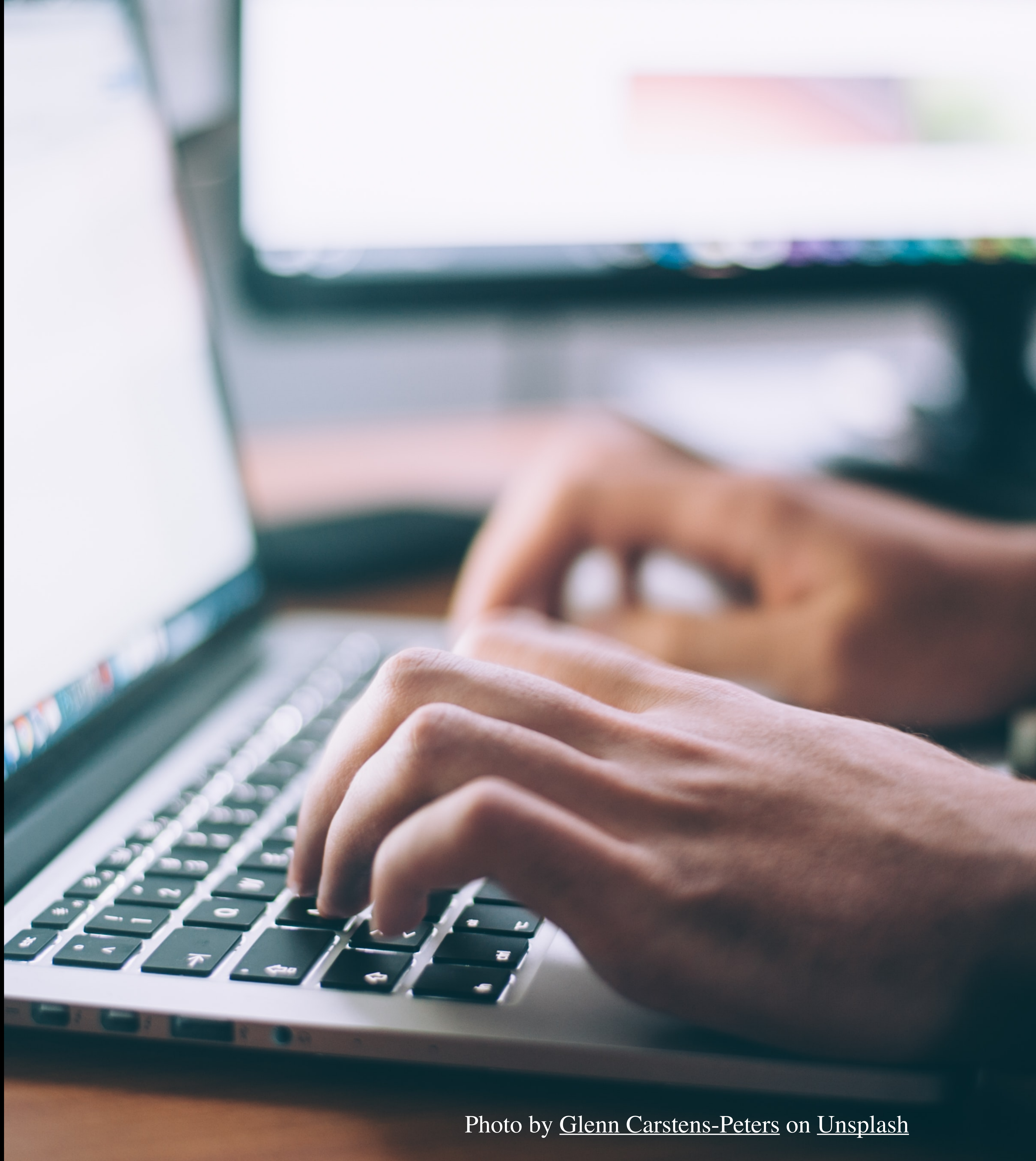


Photo by Glenn Carstens-Peters on Unsplash

METHODOLOGY

- Multiclass Classification [‘retirement', 'protection', 'investment', 'health', ‘savings']
    - GradientBoostingClassifier, RandomForestClassifier
- Features
    - ['health_insured_amt_acc', 'investment_insured_amt_acc', 'protection_insured_amt_acc', 'retirement_insured_amt_acc', 'savings_insured_amt_acc', 'ul_insured_amt_acc', 'self_insured_amt_acc', ‘dependent_insured_amt_acc']
    - [‘health_anp_acc', 'investment_anp_acc', 'protection_anp_acc', 'retirement_anp_acc', 'savings_anp_acc', 'ul_anp_acc', 'self_anp_acc', ‘dependent_anp_acc’]
    - [‘lapsed_acc', 'surrendered_acc', 'inforce_acc', 'others_acc', 'cancelled_acc', ‘claimed_acc'],
    - [‘policy_count', 'cust_tenure_months', 'cust_recency_months', 'first_product_cat', 'latest_product_cat', 'owner_age_grp', 'owner_gender', 'owner_occupation_grp', 'region_dummy']

MODEL METRICS

Pipeline

1) Preprocessor: transform features, features selection

2) Classifier: GradientBoostingClassifier, RandomForestClassifier

```python
pipe = Pipeline(steps=[('full_preprocessor', feat_select_pipe),
                       ('clf',GradientBoostingClassifier())]) #dummy

k_fold = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)

# Hyperparameters

param_grid = [{'clf':[GradientBoostingClassifier()],
            'clf__n_estimators': [100],
            'clf__max_depth': [3,4,5],
            'clf__min_samples_leaf': [2,3],
            'clf__learning_rate': [0.1,0.05,0.03,0.01]
            },
            {'clf':[RandomForestClassifier()],
            'clf__n_estimators': [500],
            'clf__max_depth': [4,5,6],
            'clf__max_features':['sqrt'],
            'clf__min_samples_leaf': [2,3]}]

best_model = GridSearchCV(pipe, param_grid = param_grid, scoring='accuracy',
                          cv = k_fold, n_jobs = -1, verbose = 1)
```

Pipeline

Metrics

```
classifier: GradientBoostingClassifier(learning_rate=0.01, max_depth=4, min_samples_leaf=2)
Model Accuracy: 0.751
Precision of postive cases: 0.75
Recall of postive cases: 0.75
F1 Score of postive cases: 0.74
```