project

glong Jia, Himanshu Gupta

Wayfair ecommerce

Overview

Today, the home furniture market has exploded to over $600 billion in the U.S and Europe, and Wayfair is the largest company in the space. Wayfair understood that home goods e-commerce required a different buyer experience than other retail goods, and developed proprietary data science and visualization technologies specifically suited towards meeting this need.

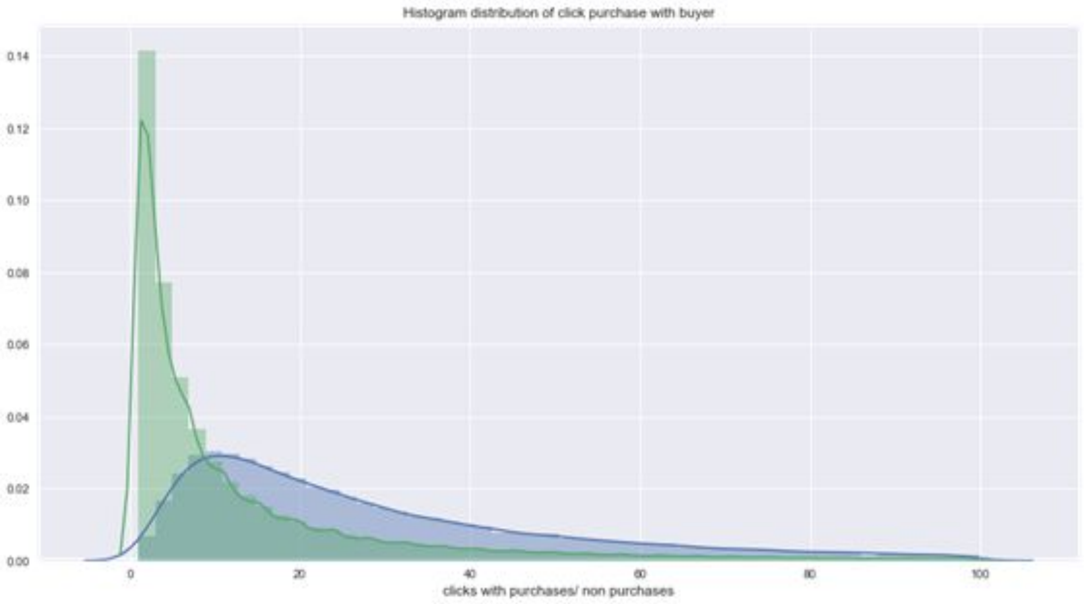
Goal

To analyze the Wayfair ecommerce clickstream data, potentially in combination with supplementary datasets, in order to increase the understanding of how various factors influence purchasing patterns on the Wayfair online platform

EDA

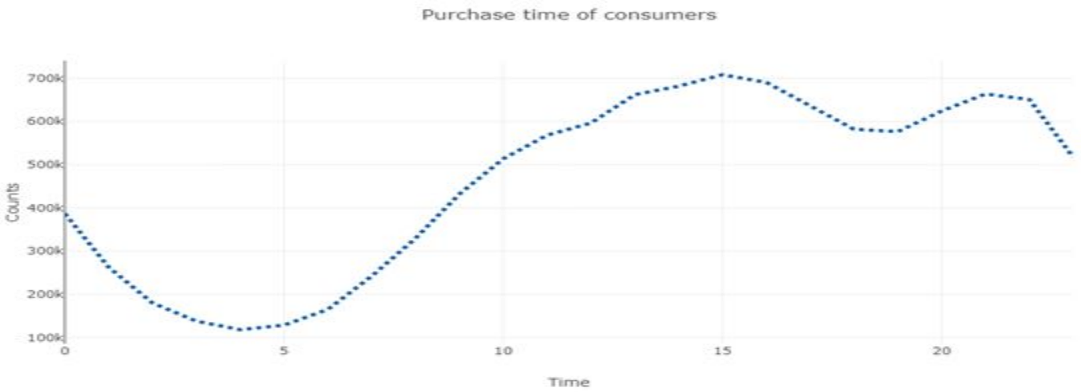
Clickthrough Buyers v/s Non-Buyers

1. Buyers clicks more on an average than non- buyers



BUYING HABITS

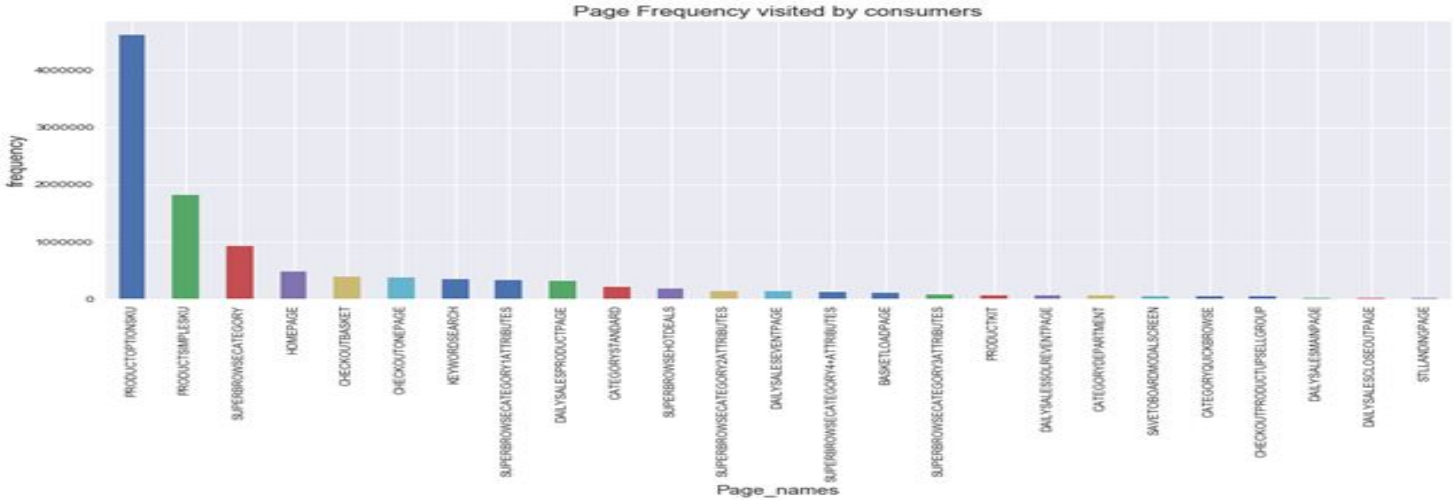
1. Customers clicks peaks at 3 pm to 11 pm , means after office hours.



CONSUMER BEHAVIOUR USING PAGE VIEWS:

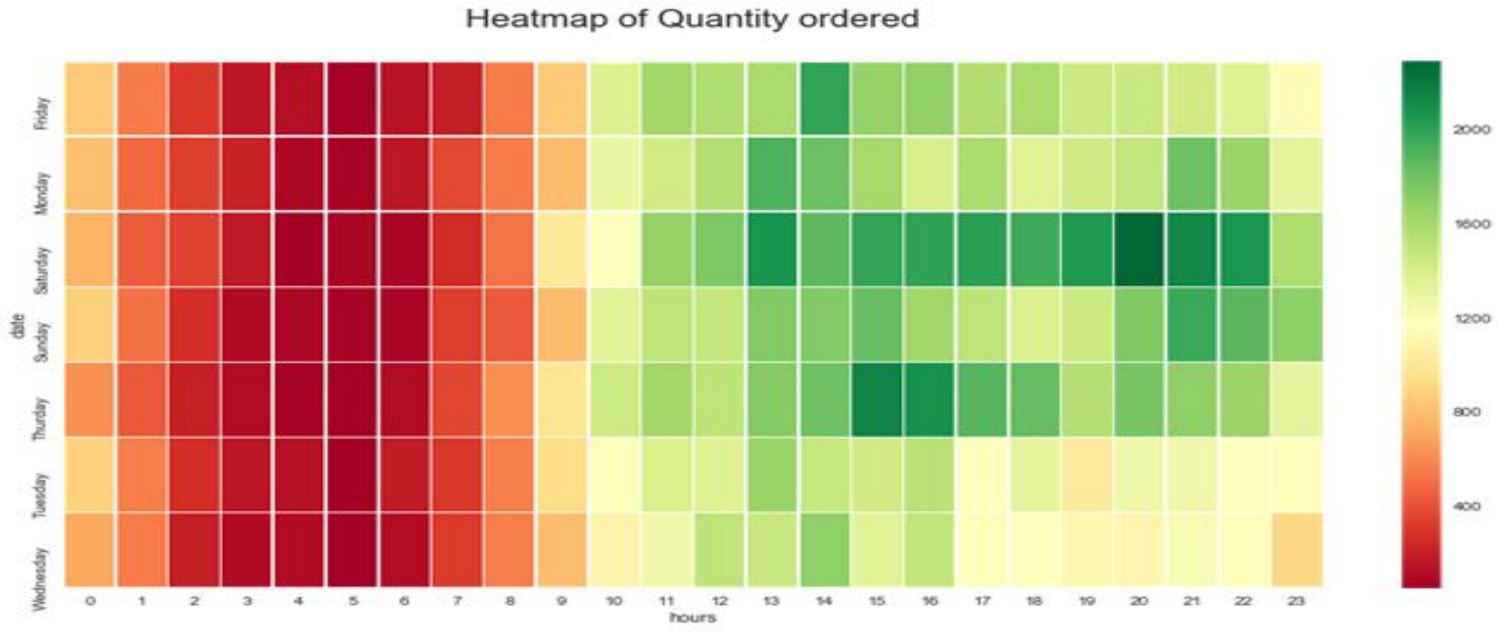
Insights: .People have clicked Products options more often than HotDeals page.

Reasons: It means two things either the people are not concerned in paying more money for paying when buying from Wayfair. or the Hot Deals page deals does not attract customers.



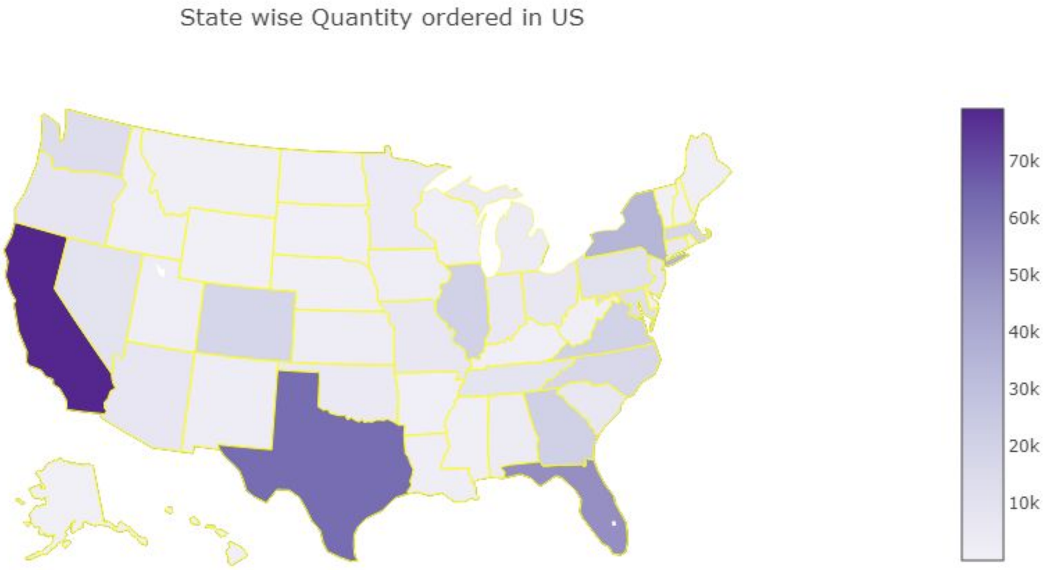
BEST DAY AND TIME FOR ORDERING BASED ON QUANTITY?

1. Saturday is the best day for ordering and most appropriate time is 8 pm.
2. 1 pm to 11 pm is prefered time for ordering.
3. Friday and Thursday are second best days to order.



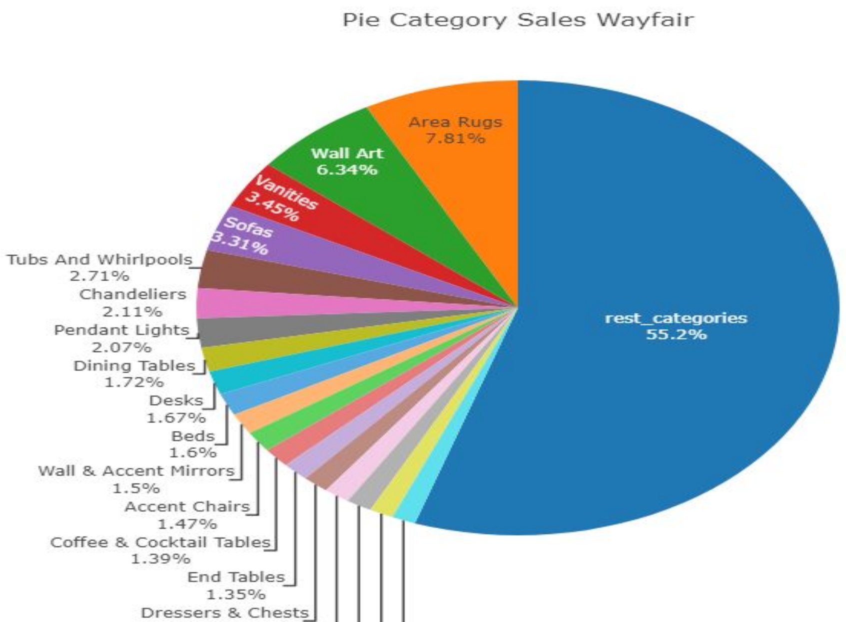
STATES PREFER BUYING FROM WAYFAIR?

1. People orders maximum quantities on Saturday and time varies from 3pm to 10 pm.

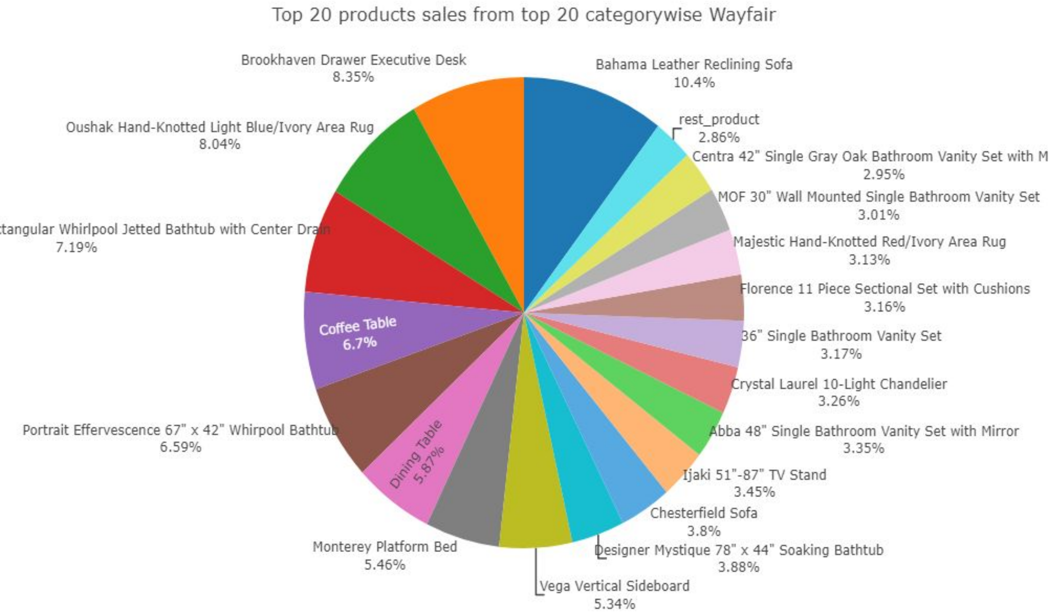


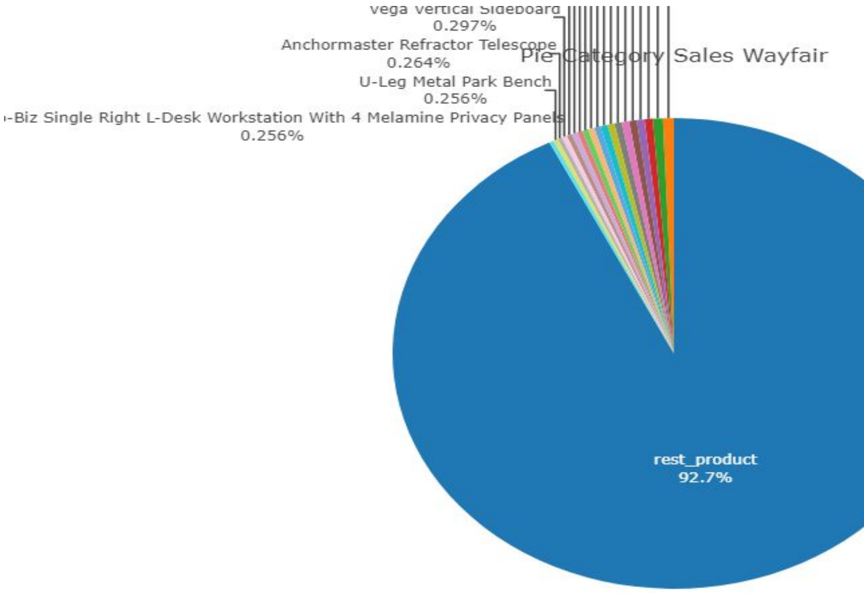
WAYFAIR TOP 20 CATEGORIES SALESWISE

1. Area Rugs, Wall Art, Vanities, Sofas, Tubs & Whirlpools are the top 5 categories.
2. Wayfair sells products in almost 900 categories. From the pie chart top 20 categories sales contributes the 45% of the sales and rest 880 categories contributes the rest 55% of the sales.



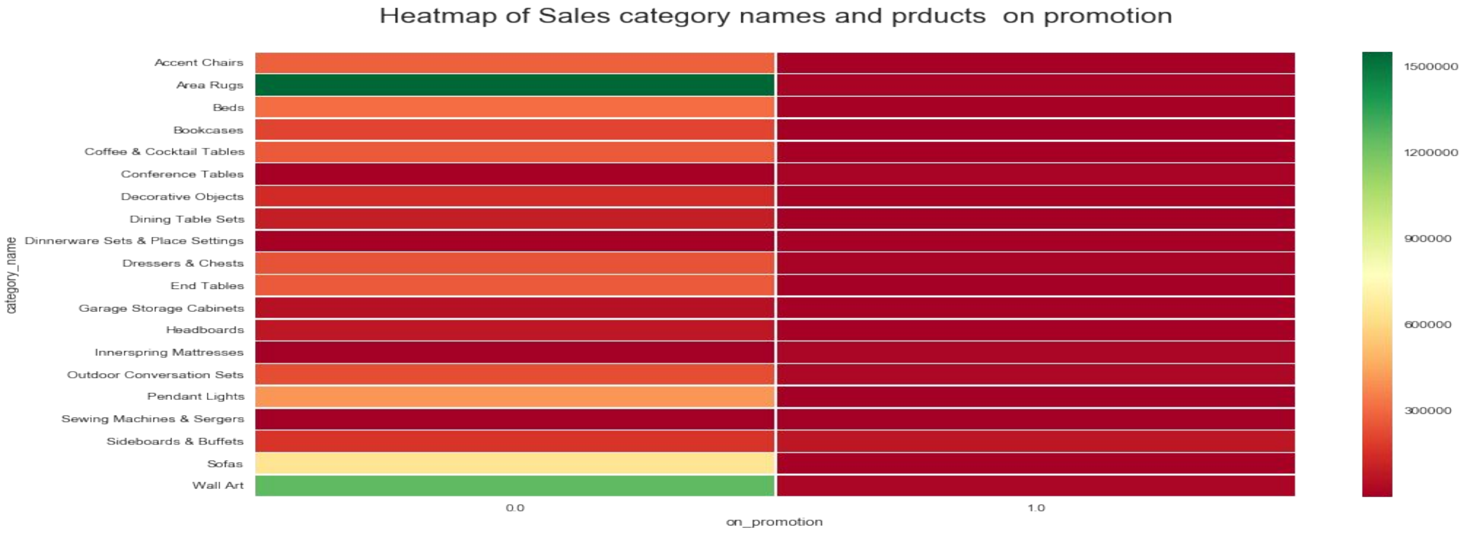
Top 20 Product overall v/s Top 20 products in Top 20 categories





CATEGORY SALES BASED ON PROMOTION

1. Products which were top sellings little less in demand during non promotion..
2. Wall Arts and Area Rugs looks outliers.
3. Other products maintained the trends of selling heavily during promotion offers.



DOES REVIEWS IMPACT SALES

MOST RELEVANT WORDS IN PRODUCT DESCRIPTION

Classification - Data modification

- Create a new target variable (final_ordered_total): For the data in the table clickstream_with_purchase, set to 1, and for the data in the able clickstream_without_purchase, set to 0.
- Dummy variables:
  - Since we want to know the customer behavior pattern on clickstream, we group by [customer_id, date, time] and get the aggregate sum for individual web pages
  -

CATEGORYDEPARTMENT_total	CATEGORYQUICKBROWSE_total	CATEGORYSTANDARD_total	CHECKOUTBASKET_total	CHECKOUTPRODUCTUPSELLGROUP_total
0	0	0	0	0
0	0	0	0	0
0	0	0	1	0
0	0	0	0	0
0	0	0	1	0

- Date variable: ‘2018-07-16’ -> ‘Monday’/ ’Tuesday’
- Time variable: ‘17:12:15’ -> ‘afternoon’ / ’morning‘...

Data Modification

Further modification for the data when we find out one variable ‘CHECKOUTONEPAGE’ plays a huge role in our model and the result depends too much on it.

We remove the column ‘CHECKOUTONEPAGE’

variable	relative_importance	scaled_importance	percentage
CHECKOUTONEPAGE	27.7706928	1.0	0.4514441
C1	18.2082996	0.6556660	0.2959965
date	6.1458783	0.2213081	0.0999082
BASKETLOADPAGE	6.0253868	0.2169693	0.0979495
time	2.5136795	0.0905155	0.0408627

Prediction on customer behavior

- Classification : to see whether there is a pattern for customer to click on different page, the time they click the web pages relate to the final purchase
- Logistic Regression
- Neural Network

Logistic Regression:

```
MAE:
0.44954942335527853
RMS:
0.6704844691380096
R2:
-0.8004232943933927
```

Neural Network:

```
MAE:
0.4462809917355372
RMS:
0.66804265712268 5
R2:
-0.787333386769012
```

- The negative r ^2 suggests both models do not fit our data
- Thus, we move to autoML tool to see which model is going to work better for our data set

AutoML for classification (H2O)

model_id	auc	logloss	mean_per_class_error	rmse	mse
StackedEnsemble_AllModels_AutoML_20181214_163440	0.587275	0.677306	0.485329	0.492214	0.242275
StackedEnsemble_BestOfFamily_AutoML_20181214_163440	0.587245	0.6774	0.487067	0.492265	0.242325
DeepLearning_1_AutoML_20181214_163440	0.580965	0.679991	0.490057	0.493574	0.243615
GBM_5_AutoML_20181214_163440	0.578229	0.682007	0.493355	0.494469	0.244499
GBM_3_AutoML_20181214_163440	0.577222	0.682502	0.491971	0.494697	0.244725
XGBoost_2_AutoML_20181214_163440	0.576828	0.682651	0.489376	0.494798	0.244825
GBM_2_AutoML_20181214_163440	0.575995	0.682258	0.490701	0.494595	0.244624
GBM_4_AutoML_20181214_163440	0.575913	0.68318	0.491451	0.495001	0.245026
XGBoost_3_AutoML_20181214_163440	0.575291	0.682743	0.494113	0.494844	0.24487
GBM_1_AutoML_20181214_163440	0.574969	0.681998	0.490776	0.494485	0.244515
XGBoost_1_AutoML_20181214_163440	0.573718	0.682478	0.490138	0.494722	0.24475
GLM_grid_1_AutoML_20181214_163440_model_1	0.573529	0.681887	0.486574	0.494446	0.244477
XGBoost_grid_1_AutoML_20181214_163440_model_3	0.572595	0.682723	0.492202	0.49485	0.244876
XGBoost_grid_1_AutoML_20181214_163440_model_4	0.572413	0.682764	0.491705	0.49486	0.244887
DRF_1_AutoML_20181214_163440	0.569304	0.690522	0.496793	0.497719	0.247724
GBM_grid_1_AutoML_20181214_163440_model_1	0.568977	0.686565	0.499322	0.496531	0.246543
XGBoost_grid_1_AutoML_20181214_163440_model_1	0.567786	0.683797	0.490553	0.495375	0.245397
XGBoost_grid_1_AutoML_20181214_163440_model_2	0.566285	0.688393	0.496494	0.497345	0.247352
DeepLearning_grid_1_AutoML_20181214_163440_model_2	0.564803	0.684068	0.492958	0.495495	0.245515
DeepLearning_grid_1_AutoML_20181214_163440_model_1	0.56024	0.686472	0.49726	0.496249	0.246263
GBM_grid_1_AutoML_20181214_163440_model_2	0.556593	0.695436	0.496049	0.500293	0.250293
XRT_1_AutoML_20181214_163440	0.545482	0.691774	0.499073	0.499257	0.249258

```
Model Details
=============
H2OStackedEnsembleEstimator :  Stacked Ensemble
Model Key:  StackedEnsemble_AllModels_AutoML_20181214_163440
No model summary for this model


ModelMetricsBinomialGLM: stackedensemble
** Reported on train data. **

MSE: 0.23613513919900206
RMSE: 0.4859373819732354
LogLoss: 0.6643520954584566
Null degrees of freedom: 77276
Residual degrees of freedom: 77266
Null deviance: 107044.0459426419
Residual deviance: 102678.27376148633
AIC: 102700.27376148633
AUC: 0.6340554090181452
pr_auc: 0.6260412344884925
Gini: 0.26811081803629033
Confusion Matrix (Act/Pred) for max f1 @ threshold = 0.38153543526596273:
```

	0	1	Error	Rate
0	4961.0	34956.0	0.8757	(34956.0/39917.0)
1	1595.0	35765.0	0.0427	(1595.0/37360.0)
Total	6556.0	70721.0	0.473	(36551.0/77277.0)

AutoML for classification (Tpot)

```python
import numpy as np
import pandas as pd
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split

# NOTE: Make sure that the class is labeled 'target' in the data file
tpot_data = pd.read_csv('PATH/TO/DATA/FILE', sep='COLUMN_SEPARATOR', dtype=np.float64)
features = tpot_data.drop('target', axis=1).values
training_features, testing_features, training_target, testing_target = \
            train_test_split(features, tpot_data['target'].values, random_state=None)

# Average CV score on the training set was:0.5575320803322882
exported_pipeline = GradientBoostingClassifier(learning_rate=0.5, max_depth=3, max_features=0.7500000000000001,
min_samples_leaf=12, min_samples_split=10, n_estimators=100, subsample=0.9500000000000001)

exported_pipeline.fit(training_features, training_target)
results = exported_pipeline.predict(testing_features)
```

SALES PREDICTION

STEPS IN SALES PREDICTION

1. DATA CLEANING & PREPROCESSING
2. FEATURE ENGINEERING
3. VARIABLE SELECTION
4. MODELLING
5. TESTING

DATA CLEANING, PREPROCESSING AND CLEANING

DATASETS USED: WAYFAIR PRODUCTS, ORDERS AND TAXES.

DATA CLEANING: Joined all three datasets based on PRODUCT_ID AND ZIPCODE AND remove all NA values from dataset..

FEATURE ENGINEERING: After joining we have 21 variables. Further we created 5 more variables to make model more effective in prediction.

1. CATEGORY RATING Based on sales.
2. MANUFACTURER RATING based on sales
3. DESCRIPITION RATING based on high frequency words in product description
4. Converted time into Morning, Afternoon, Evening, Night.
5. Converted date into Weekdays.
6. Created NSM to signify the number of seconds from midnight.

MODEL FINAL PREDICTORS: quantity_ordered, weight, num_reviews, onsite_price, on_promotion, num_return, NSM, Friday, Monday, Saturday, Sunday, Thursday, Tuesday, Wednesday, Afternoon, Evening, Morning, Night, description_length, category_count, manu_count, category_rating, manufacturer_rating, index, Desc_rate.

AUTOML H2O

	model_id	mean_residual_deviance	rmse	mse	mae	rmsle
0	StackedEnsemble_AllModels_AutoML_20181214_211548	1813.145111	42.581042	1813.145111	10.344102	NaN
1	StackedEnsemble_BestOfFamily_AutoML_20181214_2...	1813.145111	42.581042	1813.145111	10.344102	NaN
2	DRF_1_AutoML_20181214_211548	2130.669414	46.159175	2130.669414	4.590798	0.045098
3	DRF_1_AutoML_20181214_212647	2275.367753	47.700815	2275.367753	4.737439	0.045384
4	XRT_1_AutoML_20181214_212647	2495.287995	49.952858	2495.287995	5.665747	0.051432
5	XRT_1_AutoML_20181214_211548	2495.287995	49.952858	2495.287995	5.665747	0.051432
6	GLM_grid_1_AutoML_20181214_212647_model_1	374699.407358	612.126954	374699.407358	262.615097	1.515323
7	GLM_grid_1_AutoML_20181214_211548_model_1	374699.407358	612.126954	374699.407358	262.615097	1.515323

AutoML for regression prediction

Tpot model

```
Generation 1 - Current best internal CV score: -5765.543636130892
Generation 2 - Current best internal CV score: -4982.750818925716
Generation 3 - Current best internal CV score: -4982.750818925716
Generation 4 - Current best internal CV score: -2639.0953017832867
Generation 5 - Current best internal CV score: -2639.0953017832867

Best pipeline: RandomForestRegressor(MaxAbsScaler(input_matrix), bootstrap=False, max_features=0.9000000000000001, mi
n_samples_leaf=1, min_samples_split=3, n_estimators=100)
-32309.805948525587
```

```
Model Details
=============
H2OXGBoostEstimator :  XGBoost
Model Key:  XGBoost_1_AutoML_20181214_145923


ModelMetricsRegression: xgboost
** Reported on train data. **

MSE: 254.3326720456985
RMSE: 15.947810885688936
MAE: 5.940524336998396
RMSLE: NaN
Mean Residual Deviance: 254.3326720456985

ModelMetricsRegression: xgboost
** Reported on validation data. **

MSE: 9537.443917306739
RMSE: 97.65983779070463
MAE: 9.057374964284229
RMSLE: 0.08488215539378789
Mean Residual Deviance: 9537.443917306739

ModelMetricsRegression: xgboost
** Reported on cross-validation data. **

MSE: 4961.4838295815725
RMSE: 70.43780114101783
MAE: 8.260402141480078
RMSLE: NaN
Mean Residual Deviance: 4961.4838295815725
Cross-Validation Metrics Summary:
```

LINEAR REGRESSION

Training

```
#Training dataset:
lm = LinearRegression()
lm.fit(X_train, y_train)
train_pred = lm.predict(X_train)
```

```
#Mean Absolute Error:
#RMSE:
#MAPE:
print("MAE:" + str(mean_absolute_error(y_train, train_pred)))
print("RMS: " + str(sqrt(mean_squared_error(y_train, train_pred))))
print("MAPE: " + str(mean_absolute_percentage_error(y_train, train_pred)))
```

```
MAE:59.41379021136503
RMS: 296.25327549722067
MAPE: 55.74466065034095
```

Testing:

```
test_pred = lm.predict(X_test)
```

```
print("MAE:" + str(mean_absolute_error(y_test, test_pred)))
print("RMS: " + str(sqrt(mean_squared_error(y_test, test_pred))))
print("MAPE: " + str(mean_absolute_percentage_error(y_test, test_pred)))
```

```
MAE:58.73361460866351
RMS: 288.3207785371299
MAPE: 54.88796171507976
```

RANDOM FOREST

Training ¶

```
clf = RandomForestRegressor(n_estimators= 1000, random_state= 42)c
lf.fit(X_train, y_train)
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_split=1e-07, min_samples_leaf=1,
           min_samples_split=2, min_weight_fraction_leaf=0.0,
           n_estimators=1000, n_jobs=1, oob_score=False, random_state=42,
           verbose=0, warm_start=False)
```

```
train_predict_rf = clf.predict(X_train)
print("MAE:" + str(mean_absolute_error(y_train, train_predict_rf)))
print("RMS: " + str(sqrt(mean_squared_error(y_train, train_predict_rf))))
print("MAPE: " + str(mean_absolute_percentage_error(y_train, train_predict_rf)))
```

```
MAE:0.6508129517798722
RMS: 31.990554348956703
MAPE: 0.024140004005520327
```

Testing

```
test_predict_rf = clf.predict(X_test)

print("MAE:" + str(mean_absolute_error(y_test, test_predict_rf)))
print("RMS: " + str(sqrt(mean_squared_error(y_test, test_predict_rf))))
print("MAPE: " + str(mean_absolute_percentage_error(y_test, test_predict_rf)))
```

```
MAE:1.1473407420343684
RMS: 47.688213027304904
MAPE: 0.04146995207964064
```

NEURAL NETWORKS (USING KERAS)

```
train_pred_ann = classifier.predict(X_train)
```

```
print("MAE:" + str(mean_absolute_error(y_train, train_pred_ann)))
print("RMS: " + str(sqrt(mean_squared_error(y_train, train_pred_ann))))
print("MAPE: " + str(mean_absolute_percentage_error(y_train, train_pred_ann)))
```

```
MAE:215.69014876955555
RMS: 651.1622014027478
```

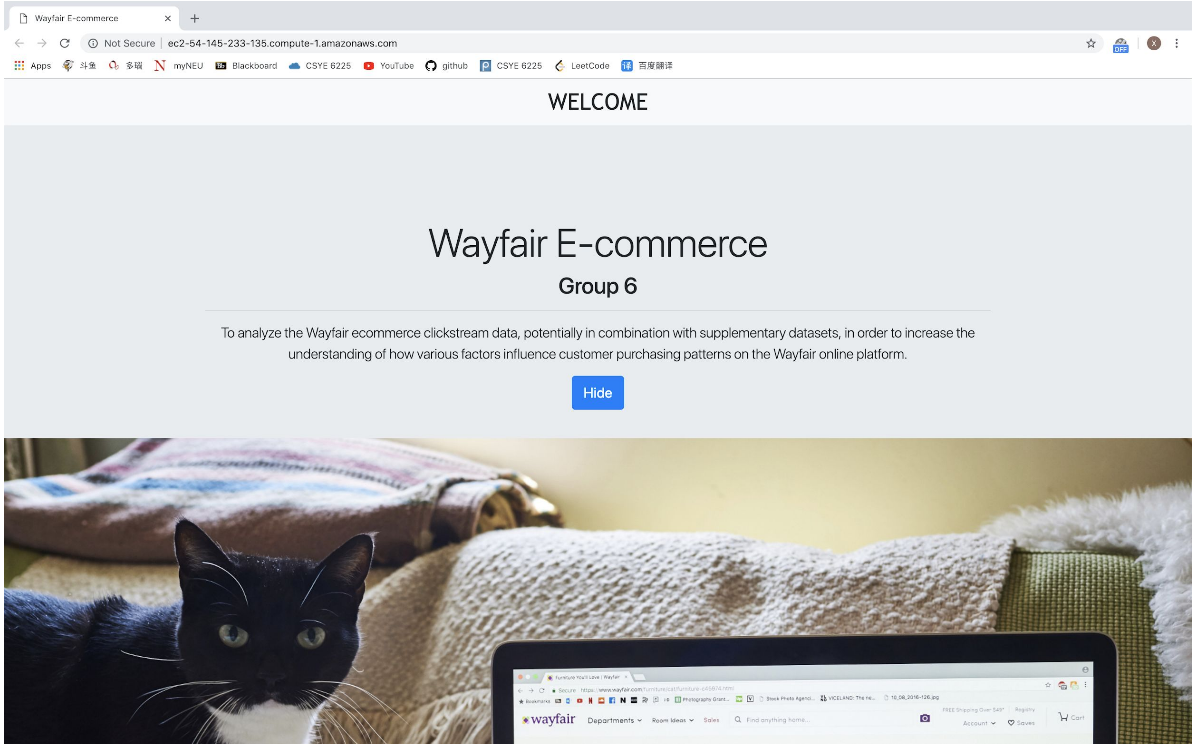
Testing

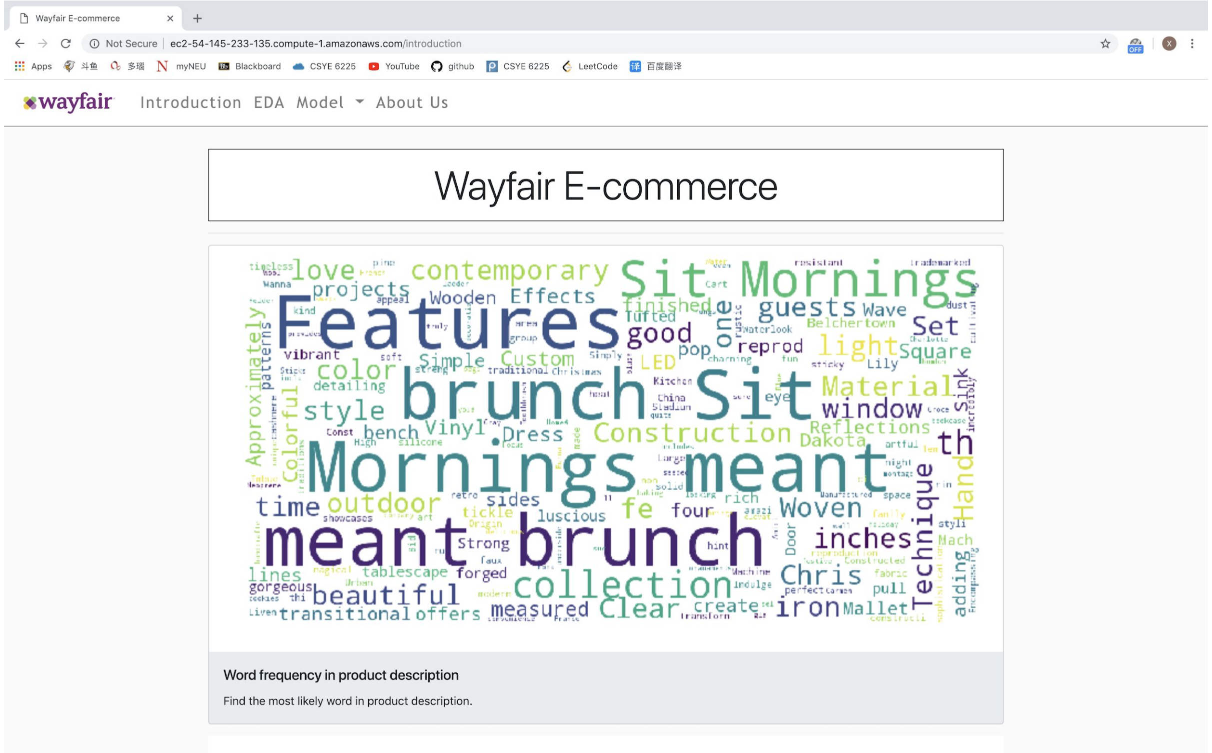
```
pred_test_ann = classifier.predict(X_test)
```

```
print("MAE:" + str(mean_absolute_error(y_test, pred_test_ann)))
print("RMS: " + str(sqrt(mean_squared_error(y_test, pred_test_ann))))
print("MAPE: " + str(mean_absolute_percentage_error(y_test, pred_test_ann)))
```

```
MAE:211.70253788653127
RMS: 628.1829339995386
MAPE: 75.10450877591661
```

Flask


Wayfair E-commerce
Not Secure | ec2-54-145-233-135.compute-1.amazonaws.com
Apps 斗鱼 多瑙 myNEU Blackboard CSYE 6225 YouTube github CSYE 6225 LeetCode 百度翻译
WELCOME
Wayfair E-commerce
Group 6
To analyze the Wayfair ecommerce clickstream data, potentially in combination with supplementary datasets, in order to increase the understanding of how various factors influence customer purchasing patterns on the Wayfair online platform.
Hide
wayfair
Departments
Room Ideas


Wayfair E-commerce
Not Secure | ec2-54-145-233-135.compute-1.amazonaws.com/introduction
Apps 斗鱼 多瑙 myNEU Blackboard CSYE 6225 YouTube github CSYE 6225 LeetCode 百度翻译
wayfair
Introduction EDA Model About Us
Wayfair E-commerce
Word frequency in product description
Find the most likely word in product description.

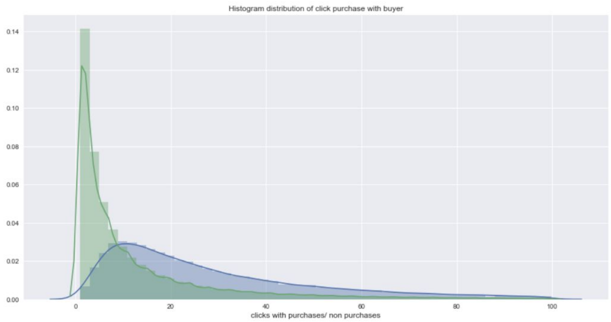
Wayfair E-commerce

Not Secure | ec2-54-145-233-135.compute-1.amazonaws.com/EDA

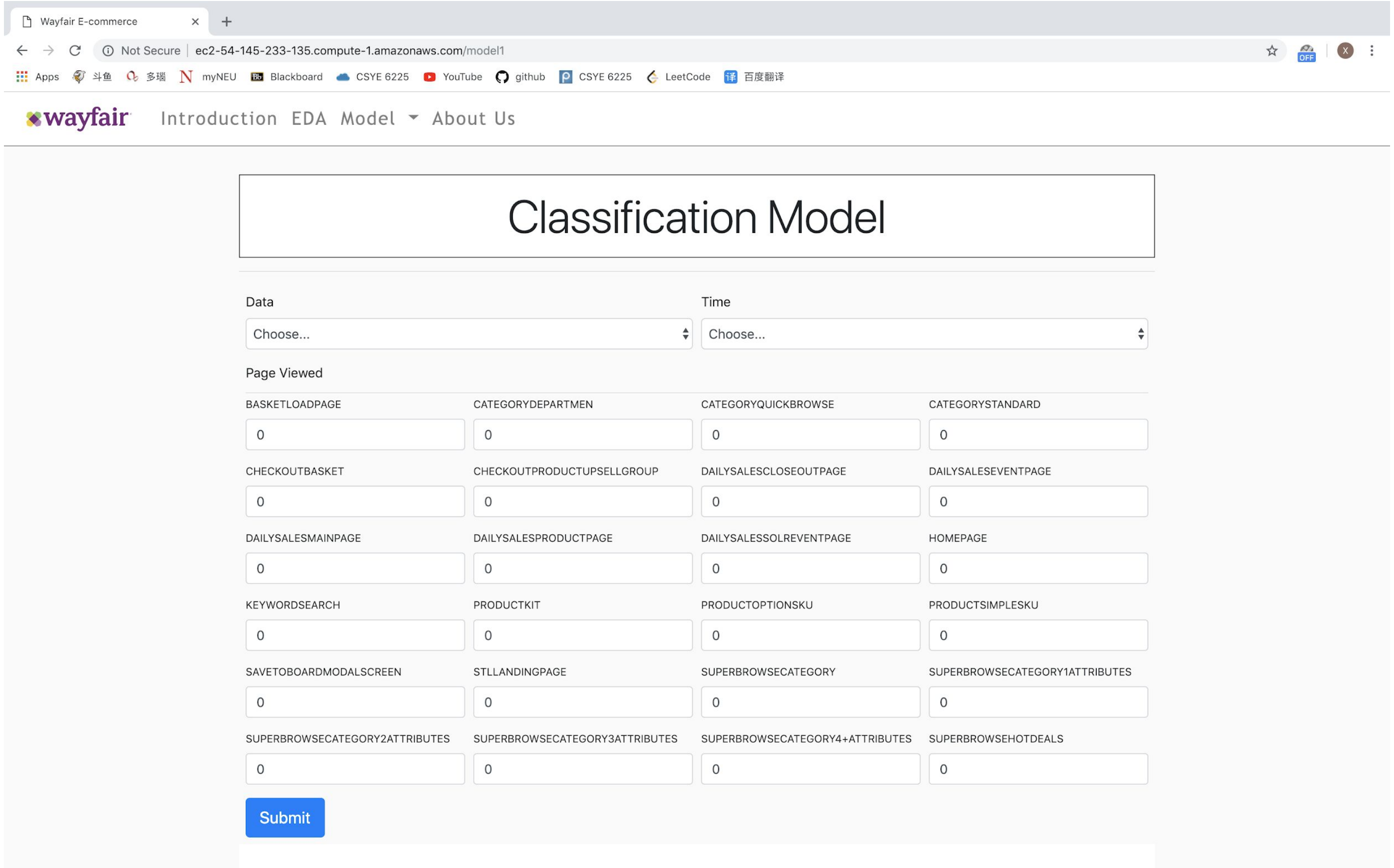
Apps 斗鱼 多瑙 myNEU Blackboard CSYE 6225 YouTube github CSYE 6225 LeetCode 百度翻译

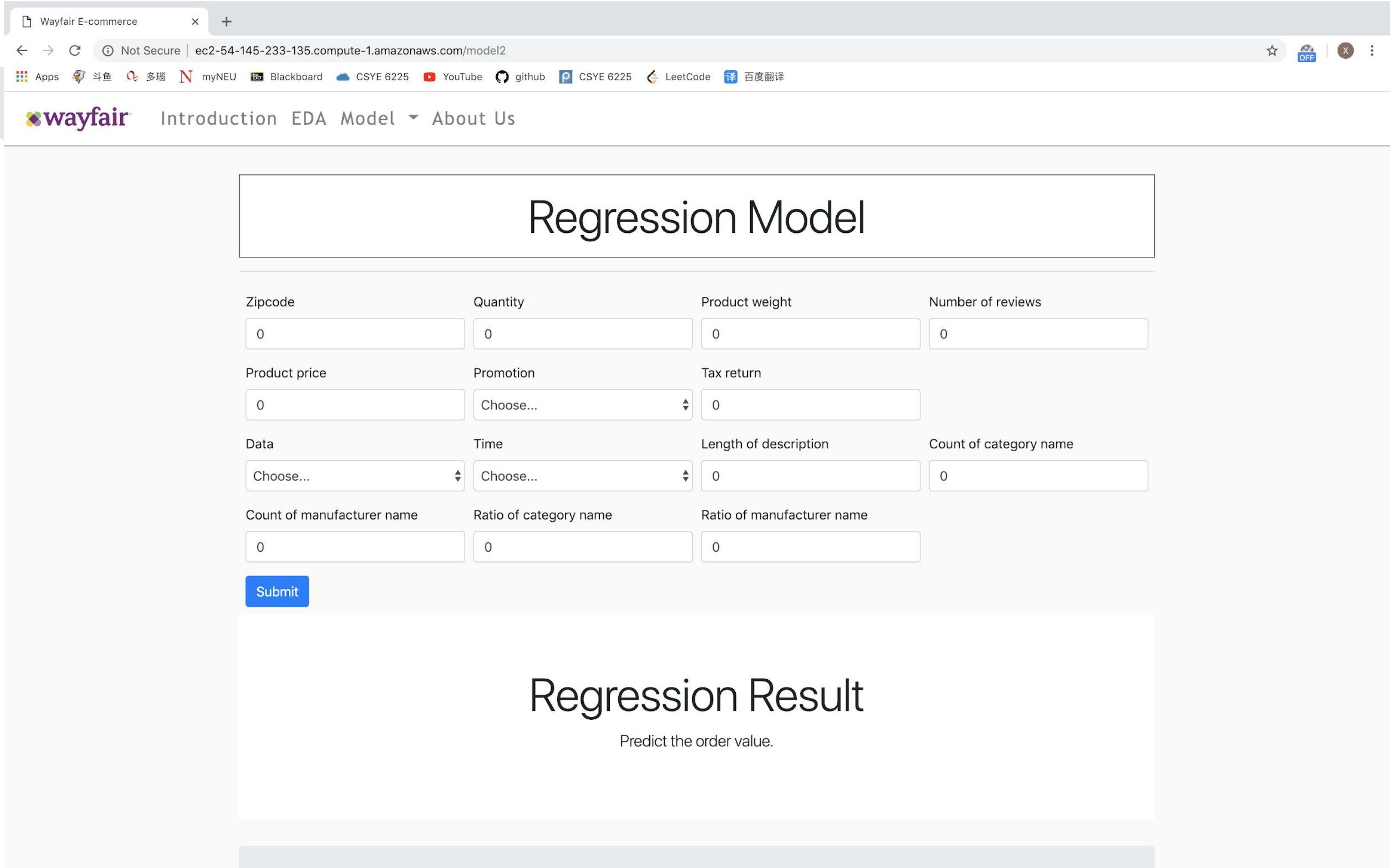
wayfair Introduction EDA Model About Us

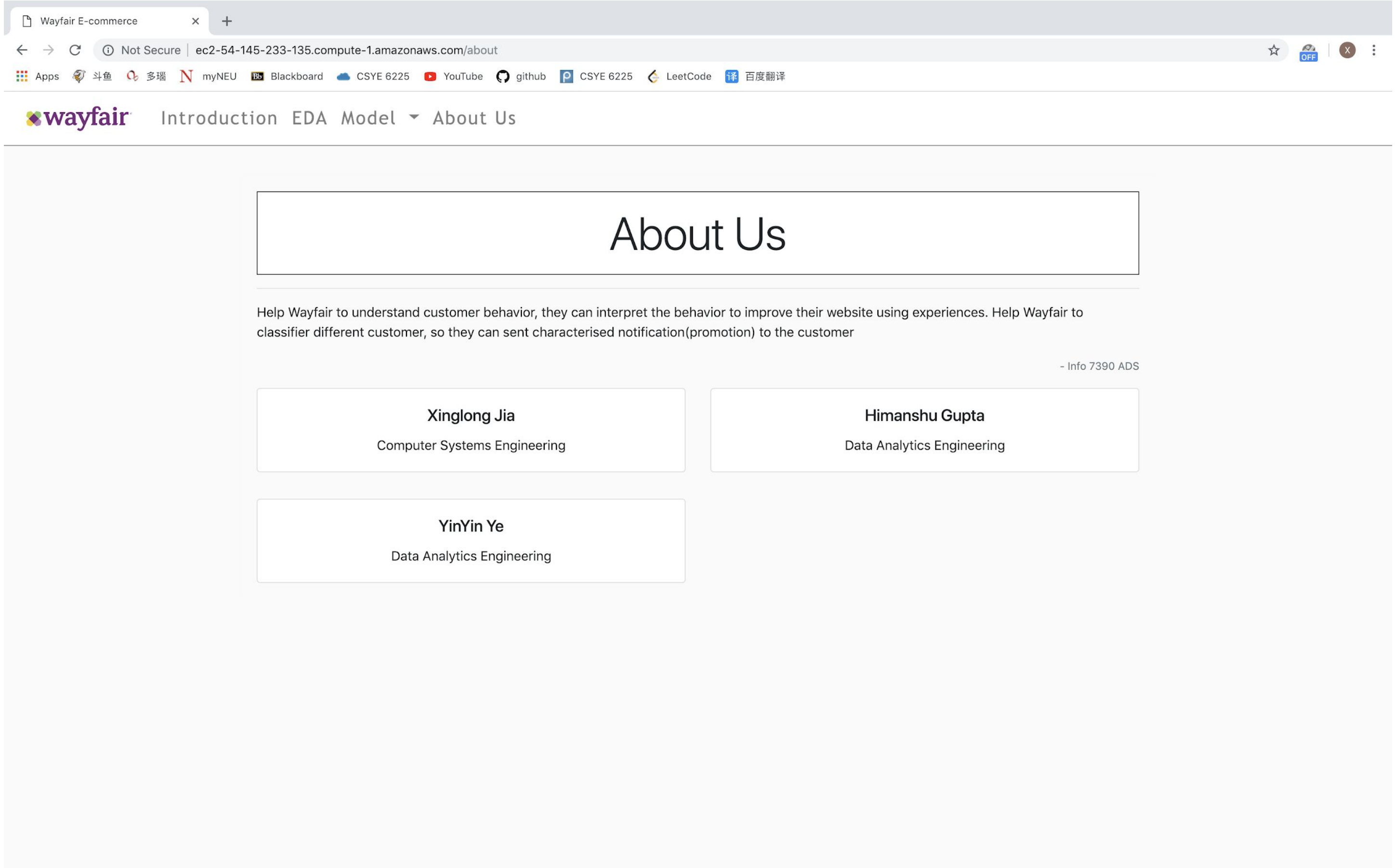
EDA



Stream with purchases/non-purchases


Wayfair E-commerce
Not Secure | ec2-54-145-233-135.compute-1.amazonaws.com/model1
Apps 斗鱼 多瑙 myNEU Blackboard CSYE 6225 YouTube github CSYE 6225 LeetCode 百度翻译
wayfair
Introduction EDA Model About Us
Classification Model
Data
Choose...
Time
Choose...
Page Viewed
BASKETLOADPAGE
0
CATEGORYDEPARTMEN
0
CATEGORYQUICKBROWSE
0
CATEGORYSTANDARD
0
CHECKOUTBASKET
0
CHECKOUTPRODUCTUPSELLGROUP
0
DAILYSALESCLOSEOUTPAGE
0
DAILYSALESEVENTPAGE
0
DAILYSALESMAINPAGE
0
DAILYSALESPRODUCTPAGE
0
DAILYSALESSOLREVENTPAGE
0
HOMEPAGE
0
KEYWORDSEARCH
0
PRODUCTKIT
0
PRODUCTOPTIONSKU
0
PRODUCTSIMPLESKU
0
SAVETOBOARDMODALSCREEN
0
STLLANDINGPAGE
0
SUPERBROWSECATEGORY
0
SUPERBROWSECATEGORY1ATTRIBUTES
0
SUPERBROWSECATEGORY2ATTRIBUTES
0
SUPERBROWSECATEGORY3ATTRIBUTES
0
SUPERBROWSECATEGORY4+ATTRIBUTES
0
SUPERBROWSEHOTDEALS
0
Submit


Wayfair E-commerce
Not Secure
ec2-54-145-233-135.compute-1.amazonaws.com/model2
Apps 斗鱼 多瑙 myNEU Blackboard CSYE 6225 YouTube github CSYE 6225 LeetCode 百度翻译
wayfair
Introduction EDA Model About Us
Regression Model
Zipcode
0
Quantity
0
Product weight
0
Number of reviews
0
Product price
0
Promotion
Choose...
Tax return
0
Data
Choose...
Time
Choose...
Length of description
0
Count of category name
0
Count of manufacturer name
0
Ratio of category name
0
Ratio of manufacturer name
0
Submit
Regression Result
Predict the order value.



About Us

Help Wayfair to understand customer behavior, they can interpret the behavior to improve their website using experiences. Help Wayfair to classifier different customer, so they can sent characterised notification(promotion) to the customer

- Info 7390 ADS

Xinglong Jia
Computer Systems Engineering

Himanshu Gupta
Data Analytics Engineering

YinYin Ye
Data Analytics Engineering

Docker

```
FROM ubuntu

# System requirements
RUN apt-get update && apt-get install -y python3-pip

RUN pip3 install --upgrade pip

# Bundle object sources
RUN mkdir /WayfairFinal
ADD flask /WayfairFinal/flask
ADD RegressionModel.ipynb /WayfairFinal
ADD new_prediction.ipynb /WayfairFinal
ADD productsComp_products.ipynb /WayfairFinal
ADD regression_data.csv /WayfairFinal
ADD script.sh /WayfairFinal

# Install python3 modules
RUN pip3 install \
    jupyter \
    pandas \
    h2o \
    flask \
    sklearn

# start download origination and perforation data
CMD /bin/bash WayfairFinal/script.sh
```

```
echo "Welcom to Wayfair E-commerce"
echo "Group 6"

echo "Want to open notebook? y/n"
read input
if [ $input == "y" ] || [ $input == "Y" ]; then
    jupyter notebook --ip 0.0.0.0 --no-browser --allow-root --notebook-dir='/WayfairFinal'
fi

echo "Want to open web application? y/n"
read input1
if [ $input == "y" ] || [ $input == "Y" ]; then
    python3 WayfairFinal/flask/app.py
fi

cd WayfairFinal
ls
```

localhost

Regressi... | python -... | Shibbole... | CAS – Ce... | final_rep... | Final_pro... | How can... | Hello Wh... | Home

jupyter

Quit Logout

Files Running Clusters

Select items to perform actions on them.

Upload New

0 /	Name	Last Modified	File size
	flask	an hour ago	
	new_prediction.ipynb	7 hours ago	197 kB
	productsComp_products.ipynb	16 hours ago	474 kB
	RegressionModel.ipynb	2 hours ago	373 kB
	regression_data.csv	3 hours ago	9.57 MB
	script.sh	13 minutes ago	401 B