The Price is Right!

Rotem Nizhar & Batel Mankovsky

Notes on the differences between models 1, 2 & 3

In all submissions we used a Random Forest regressor after running randomized search CV, altought we tried to train several other models, as we also mentioned in our PDF summary (GB, KNN...).

Model 1:

We used most of the features that we devided were important, after much preprocessing and cleaning, but it was before we created additional features as well.
The reason the model performed poorly was because we didn't notice that for the "heel_height" feature, all observations that didn't have the word low/medium/high in their "heel_height" data were left with the original value as their heel height, instead of receiving "other" as a value. That caused our model to have many many dummy features after the encoding of categorical data, which were also of course non informative and probably caused overfitting.

Model 2:

Here we of course fixed the previous problem, and not only gave all unknown observations the category "other" for heel height, we also extracted the heel height and the unit (inch/cm/mm) from the string, which worked for a lot of observations. Then, we converted all heights to centimeters, and then divided it to 6 categories of height. We hoped this feature will somehow improve the model, but we saw only a small improvement when we checked it.

Additionaly, we extracted several words from the title and the seller notes that seemed to us like a good indication of the priciness of the shoe (and we also backed it with data of the average price with and without these words). We created a binary feature that is 1 if one of those words exists in the title/notes, and 0 otherwise.

Model 3:

The biggest change we introduced to the features in our 3rd model is that we added additional features that represent the priciness of the brands.
The first one is a numerical feature that for every brand (known brand from the train data only) with a mean log price above 4, puts it's mean log price as the value. For all other brands, and if the brand is unknown or empty, we put the mean log price of all the other brands (that are cheaper, and have a mean log price less than 4). The reason behind this feature was to give more importance to expensive brands, even if we have bery few observations of them in the train data.

Additionally, we created a feature that orders the brands by their average price, and ranks them accordingly. For unknown on never seen before brands, we give them the average rank. This is also a numerical feature that we created to strengthen the brand's importance, even if a brand appears very few times in the data.

These 2 features seemed to have a very good affect on the model, and along with a few other features we indroduced at this stage (listed below) gave us a RMSE of ~0.52.

The additional features were:

- Extracting "fancy" words from the color feature, after we saw a correlation between their appearance and the price.
- Creating a feature that counts the number of missing values for each observation, and creating 7 categories by number of missing values. The idea behind this feature was that during EDA we saw that observation we a lot of missing data tended to be more expansive.

Model 4

we tried something different and ran a NN, but probably had some configuration issue as the results on the train set were somewhat un-calibrated, and we didn't have the time to investigate.

Imports & Settings

In [1]:
```
import json
import re
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Regressors
from sklearn.ensemble import RandomForestRegressor
from lightgbm import LGBMRegressor

# Regression Metrics
from sklearn.metrics import mean_squared_error

# Model selection
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score, RepeatedKFold
from sklearn.model_selection import RandomizedSearchCV

%matplotlib inline
%run data_preparation_for_model.ipynb # <-- Our functions for processing and creating the final DFs for the model
```

Prepare Data

In [2]:
```
df_train = pd.read_csv('./data/ebay_women_shoes_train.csv')
df_train["log_price"] = np.log(df_train.price)

X_train, X_test, y_train, y_test =  prepare_train_test_data(df_train)
```

Rndomized Search CV - Random Forest

In [3]:
```
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 400, stop = 1600, num = 7)]

# Number of features to consider at every split
max_features = ['auto', 'sqrt']

# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(10, 110, num = 6)]
max_depth.append(None)

# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]

# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]

# Method of selecting samples for training each tree
bootstrap = [True, False]

# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}
```

In [4]:
```
# Rndomized Search CV - search for best hyperparameters

# First create the base model to tune
rf = RandomForestRegressor()

# Random search of parameters, using 3 fold cross validation,
# search across 100 different combinations
rf_random = RandomizedSearchCV(
    estimator = rf,
    param_distributions = random_grid,
    n_iter = 100,
    cv = 3,
    verbose=2,
    random_state=42,
    n_jobs = -1,
    scoring='neg_root_mean_squared_error'
)

# Fit the random search model
rf_random.fit(X_train, y_train)  # <-- Might take a while

print(rf_random.best_params_)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits
{'n_estimators': 600, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': 70, 'bootstrap': False}
```

In [5]:
```
# Evaluate model
y_pred = rf_random.best_estimator_.predict(X_train)
print("RMSE on training data:")
print(round(mean_squared_error(y_train, y_pred, squared=False), 4))

y_pred_test = rf_random.best_estimator_.predict(X_test)
print("RMSE on (our) test data:")
print(round(mean_squared_error(y_test, y_pred_test, squared=False), 4))
```

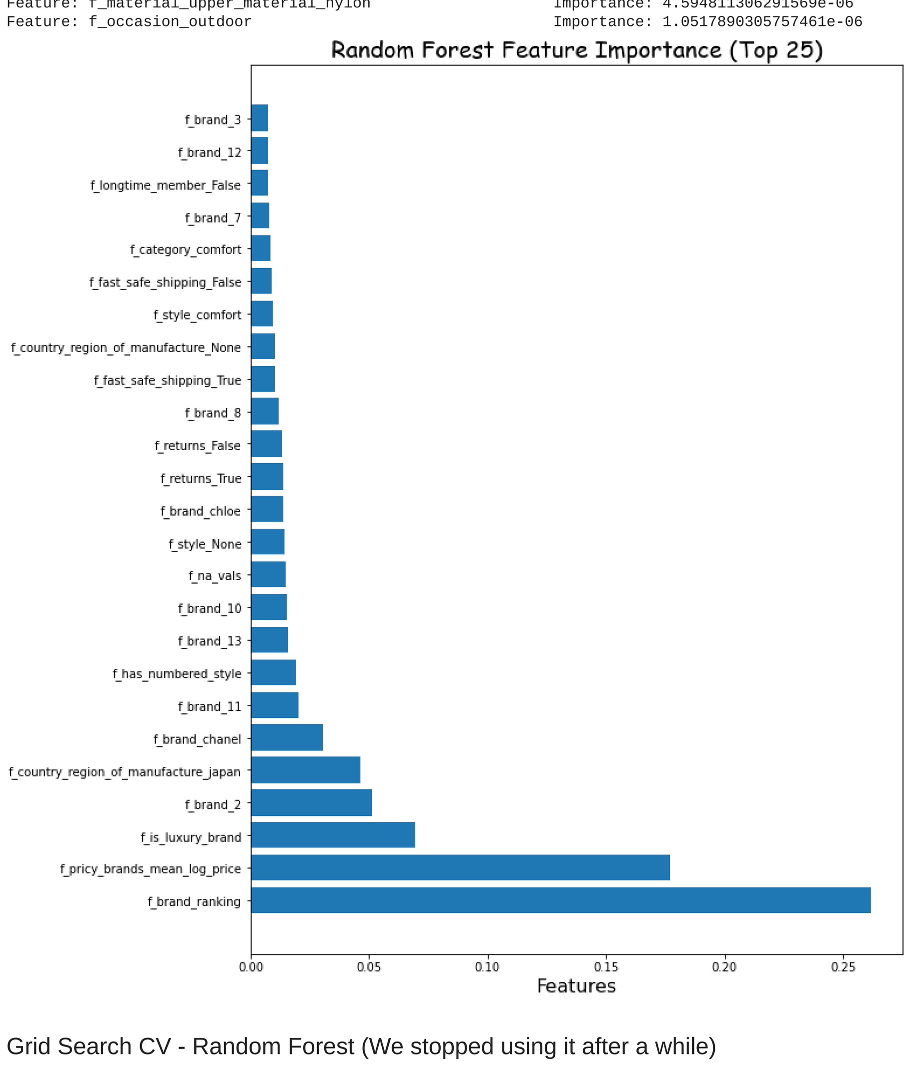
```
RMSE on training data:
0.3444
RMSE on (our) test data:
0.5137
```

In [6]:
```
# View the feature importances
feature_importances = list(zip(X_train, rf_random.best_estimator_.feature_importances_))
feature_importances_ranked = sorted(feature_importances, key = lambda x: x[1], reverse = True)

# Print out the feature and their importances
[print('Feature: {:50} Importance: {}'.format(*pair)) for pair in feature_importances_ranked];

# Plot the top 25 important features
feature_names_25 = [i[0] for i in feature_importances_ranked[:25]]
y_ticks = np.arange(0, len(feature_names_25))
x_axis = [i[1] for i in feature_importances_ranked[:25]]
plt.figure(figsize = (10, 14))
plt.barh(feature_names_25, x_axis)   #horizontal barplot
plt.title('Random Forest Feature Importance (Top 25)',
          fontdict= {'fontname':'Comic Sans MS','fontsize' : 20})
plt.xlabel('Features',fontdict= {'fontsize' : 16})
plt.show()
```

```
Feature: f_brand_ranking                                    Importance: 0.2619050982147061
Feature: f_pricy_brands_mean_log_price                      Importance: 0.17694810018674276
Feature: f_is_luxury_brand                                  Importance: 0.06960086487025656
Feature: f_brand_2                                          Importance: 0.05106499043918853
Feature: f_country_region_of_manufacture_japan              Importance: 0.046123603536946
Feature: f_brand_chanel                                     Importance: 0.030613622592224117
Feature: f_brand_11                                         Importance: 0.020258344860469638
Feature: f_has_numbered_style                               Importance: 0.01906256717668859
Feature: f_brand_13                                         Importance: 0.015635499185482182
Feature: f_brand_10                                         Importance: 0.015337572825193952
Feature: f_na_vals                                          Importance: 0.014799043349535395
Feature: f_style_None                                       Importance: 0.014301157688860398
Feature: f_brand_chloe                                      Importance: 0.013799873585669954
Feature: f_returns_True                                     Importance: 0.01368253747692065
Feature: f_returns_False                                    Importance: 0.013177126004922041
Feature: f_brand_8                                          Importance: 0.012085418386255282
Feature: f_fast_safe_shipping_True                          Importance: 0.01042720172111177
Feature: f_country_region_of_manufacture_None               Importance: 0.01038466702557676
Feature: f_style_comfort                                    Importance: 0.009212918446247605
Feature: f_fast_safe_shipping_False                         Importance: 0.009059127167710726
Feature: f_category_comfort                                 Importance: 0.008316896653131874
Feature: f_brand_7                                          Importance: 0.007702362485992487
Feature: f_longtime_member_False                            Importance: 0.007543465866133313
Feature: f_brand_12                                         Importance: 0.007461876848922508
Feature: f_brand_3                                          Importance: 0.007398119803214938
Feature: f_category_heels                                   Importance: 0.006900439060621277
Feature: f_category_flats                                   Importance: 0.006312497757108328
Feature: f_brand_4                                          Importance: 0.005739886050853135
Feature: f_material_upper_material_other                    Importance: 0.0056994070230380976
Feature: f_brand_1                                          Importance: 0.005678357040105858
Feature: f_longtime_member_True                             Importance: 0.005634368308957878
Feature: f_brand_14                                         Importance: 0.005207898760781848
Feature: f_style_Other                                      Importance: 0.004653496897241465
Feature: f_material_upper_material_leather                  Importance: 0.004590421704338355
Feature: f_brand_6                                          Importance: 0.00418628585941587
Feature: f_heel_height_Unknown                              Importance: 0.0036312006300279194
Feature: f_brand_5                                          Importance: 0.0033807771060286257
Feature: f_pricy_words                                      Importance: 0.0032470655953557204
Feature: f_same_day_shipping_False                          Importance: 0.003235325975561046
Feature: f_same_day_shipping_True                           Importance: 0.0032164105286729924
Feature: f_brand_0                                          Importance: 0.0029038028050603123
Feature: f_free_shipping_True                               Importance: 0.002873348885734368
Feature: f_free_shipping_False                              Importance: 0.002501098236953649
Feature: f_brand_dansko                                     Importance: 0.0024820913696343254
Feature: f_style_casual                                     Importance: 0.002181705914620273
Feature: f_occasion_other                                   Importance: 0.0021786019570106765
Feature: f_heel_height_Low                                  Importance: 0.0021730876133966293
Feature: f_country_region_of_manufacture_italy              Importance: 0.0021554307683256018
Feature: f_heel_height_High                                 Importance: 0.002155353434808363
Feature: f_style_more_formal                                Importance: 0.0021154939778131317
Feature: f_heel_height_Medium                               Importance: 0.002027966975223734
Feature: f_country_region_of_manufacture_china              Importance: 0.001985379505070741
Feature: f_material_upper_material_pelle                    Importance: 0.001876963187954547
Feature: f_heel_type_None                                   Importance: 0.0018647557431301462
Feature: f_vintage_null                                     Importance: 0.0018365035323699758
Feature: f_occasion_casual                                  Importance: 0.0018332229879042886
Feature: f_vintage_False                                    Importance: 0.0018069154735903261
Feature: f_heel_height_Flat                                 Importance: 0.0017754898220131553
Feature: f_brand_born                                       Importance: 0.0016277258607805976
Feature: f_brand_clarks                                     Importance: 0.0016235913467824285
Feature: f_material_upper_material_suede                    Importance: 0.0014822700742302482
Feature: f_style_flats                                      Importance: 0.0014712989335995113
Feature: f_heel_type_High/Slim                              Importance: 0.0010169254780395256
Feature: f_style_formal                                     Importance: 0.00084170271927792
Feature: f_returns_null                                     Importance: 0.0008176933273016086
Feature: f_heel_height_Very High                            Importance: 0.000760923485505537
Feature: f_brand_cole haan                                  Importance: 0.0007569164031213414
Feature: f_style_oxford                                     Importance: 0.0007358991581721363
Feature: f_material_upper_material_canvas                   Importance: 0.000691030542739909
Feature: f_style_sandals                                    Importance: 0.0006285124676538153
Feature: f_brand_skechers                                   Importance: 0.0006267287593939098
Feature: f_style_ballet                                     Importance: 0.0006230330256776635
Feature: f_heel_type_Medium/Wide                            Importance: 0.0006118766372829686
Feature: f_fancy_colors                                     Importance: 0.0005938695773692832
Feature: f_brand_sas                                        Importance: 0.000564783686677864
Feature: f_occasion_party                                   Importance: 0.0005621264341039041
Feature: f_style_heels                                      Importance: 0.000560542900393022
Feature: f_brand_sperry top sider                           Importance: 0.0005114256527372209
Feature: f_country_region_of_manufacture_Other              Importance: 0.00047173398695010644
Feature: f_brand_merrell                                    Importance: 0.0004408077523328434
Feature: f_material_upper_material_faux leather             Importance: 0.0004402449357170835
Feature: f_brand_steve madden                               Importance: 0.0004250192727390865
Feature: f_brand_nine west                                  Importance: 0.0004236651812278045
Feature: f_material_upper_material_textile                  Importance: 0.00038419557066303965
Feature: f_brand_propet                                     Importance: 0.00038040684145921976
Feature: f_country_region_of_manufacture_united states      Importance: 0.00032011848191617985
Feature: f_brand_kurt geiger                                Importance: 0.000299677129110419
Feature: f_occasion_formal                                  Importance: 0.00023534285154756056
Feature: f_pricy_words_notes                                Importance: 0.0002291633708432536
Feature: f_brand_next                                       Importance: 0.00020295882383888965
Feature: f_country_region_of_manufacture_france             Importance: 0.00020120633033467246
Feature: f_country_region_of_manufacture_spain              Importance: 0.00017501621510228083
Feature: f_occasion_any                                     Importance: 0.00017316578858828858
Feature: f_country_region_of_manufacture_unknown            Importance: 0.0001727805096562108
Feature: f_country_region_of_manufacture_brazil             Importance: 0.00016150896280152605
Feature: f_material_upper_material_rubber                   Importance: 0.00015499273986265913
Feature: f_heel_type_Other                                  Importance: 0.000143156952410824
Feature: f_material_upper_material_fabric                   Importance: 0.00012851019375820027
Feature: f_material_upper_material_faux suede               Importance: 0.00012610353312974212
Feature: f_country_region_of_manufacture_vietnam            Importance: 0.00012402654760037515
Feature: f_vintage_True                                     Importance: 0.00010970518412899263
Feature: f_material_upper_material_satin                    Importance: 0.00010188370361505908
Feature: f_occasion_dress                                   Importance: 9.905730079987505e-05
Feature: f_material_upper_material_nubuck                   Importance: 9.572711248187829e-05
Feature: f_material_upper_material_cotton                   Importance: 9.296834180306667e-05
Feature: f_material_upper_material_velvet                   Importance: 9.291365640333616e-05
Feature: f_country_region_of_manufacture_germany            Importance: 8.621017379770857e-05
Feature: f_occasion_wedding                                 Importance: 8.30405064652957e-05
Feature: f_country_region_of_manufacture_portugal           Importance: 7.277465386089623e-05
Feature: f_heel_type_Flat                                   Importance: 6.720248435549795e-05
Feature: f_material_upper_material_manmade                  Importance: 5.750285347426509e-05
Feature: f_country_region_of_manufacture_mexico             Importance: 5.29513105551183466-05
Feature: f_occasion_flat                                    Importance: 3.200999504238923e-05
Feature: f_country_region_of_manufacture_australia          Importance: 2.6834212412504643e-05
Feature: f_country_region_of_manufacture_israel             Importance: 2.6266650377546525e-05
Feature: f_material_upper_material_linen                    Importance: 2.0283501297300002e-05
Feature: f_occasion_business                                Importance: 6.402093792422129e-06
Feature: f_country_region_of_manufacture_united kingdom     Importance: 4.8343357425044935e-06
Feature: f_material_upper_material_nylon                    Importance: 4.594811306291569e-06
Feature: f_occasion_outdoor                                 Importance: 1.0517890305757461e-06
```



Grid Search CV - Random Forest (We stopped using it after a while)

Creating a Grid Search for Random Forest Regressor gridsearch = GridSearchCV(estimator=RandomForestRegressor(random_state=42), param_grid={ 'n_estimators': [50, 100, 250, 300, 400], # 'max_features': ['auto', 'sqrt', 'log2'], 'max_depth': [int(x) for x in np.linspace(10, 110, num = 11)] }, cv=5, return_train_score=False, scoring='neg_root_mean_squared_error') gridsearch.fit(X=X_train, y=y_train) pd.DataFrame(gridsearch.cv_results_).set_index('rank_test_score').sort_index()

Train RF model we got on whole data + submit

In [ ]:
```
df_train = pd.read_csv('./data/ebay_women_shoes_train.csv')
df_test = pd.read_csv('./data/ebay_women_shoes_test.csv')

# Save original ids order, for submission
test_ids = df_test.id

X_train, X_test, y_train = prepare_data_for_testing(df_train, df_test)
X_test.drop(columns=['log_price'], inplace=True)
```

In [ ]:
```
# Fit the best model we got from randomized search CV, ON ALL DATA!
rf_model_03 = rf_random.best_estimator_
rf_model_03.fit(X_train, y_train)
```

In [ ]:
```
# Predict on whole train data (just a sanity check)
y_pred_train = rf_model_03.predict(X_train)
print("RMSE on WHOLE training data (sanity check):")
print(round(mean_squared_error(y_train, y_pred_train, squared=False), 4))
```

In [ ]:
```
# Predict on test data
y_pred_test = rf_model_03.predict(X_test)
```

In [ ]:
```
results_model_03 = pd.DataFrame({"id": test_ids, "price_pred": y_pred_test})

# Save presiction to csv file
results_model_03.to_csv("model_03.csv", index=False)
```

Prediction: Gradient Boosting (We didn't use this model for submission)

define the model model = LGBMRegressor() # evaluate the model cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=42) n_scores = cross_val_score(model, X_train, y_train, scoring='neg_mean_squared_error', cv=cv, n_jobs=-1, error_score='raise') model.fit(X_train, y_train) y_pred = model.predict(X_test) print("RMSE on test data:") print(round(mean_squared_error(y_test, y_pred, squared=False), 4))params = { 'num_leaves': [7, 14, 21, 28, 31, 50, 70], 'learning_rate': [0.1, 0.03, 0.003], 'max_depth': [-1, 1, 2, 3, 4, 5, 6], 'n_estimators': [10, 50, 100, 200, 500, 1000], } grid = GridSearchCV(LGBMRegressor(random_state=42), params, scoring='neg_mean_squared_error', cv=5) grid.fit(X_train, y_train) # <-- Might take a whilegrid.best_params_# Evaluate model y_pred = grid.predict(X_train) print("RMSE on training data:") print(round(mean_squared_error(y_train, y_pred, squared=False), 4)) y_pred_test = grid.predict(X_test) print("RMSE on (our) test data:") print(round(mean_squared_error(y_test, y_pred_test, squared=False), 4))