Machine learning explanability

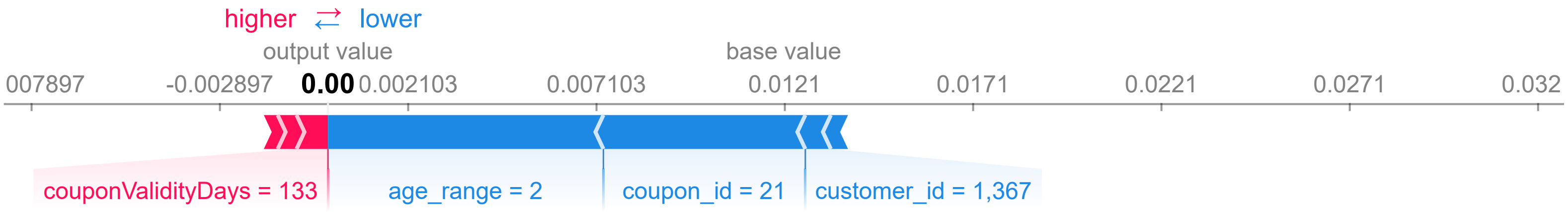
SHAP

The item's brand, customer's income and the coupon's validity are pushing up the probability of coupon redemption. But their combined effect is dwarfed by that of the features (and their values) in blue that reduce such probability, resulting in the prediction of non-redemption.

In [115]:

```
# explain the model's predictions on test set using SHAP values
# Interactive visualization of the explanation of the first subject
# in the test set (X_explain).
X_explain = X_test
X_explain.reset_index(drop=True, inplace=True)
explainer = shap.TreeExplainer(rfc2)
shap_values = explainer.shap_values(X_explain)
shap.initjs() #initialize javascript in cell
shap.force_plot(explainer.expected_value[1], shap_values[1][0,:], X_explain.iloc[0,:])
```

Out[115]:



LIME

In [0]:

```
trainX = trainCampaignCouponTransCustItem.drop(['date', 'item_id', 'quantity', 'selling_price', 'other_discou
nt', 'coupon_discount',
                                                 'redemption_status', 'campaign_type', 'start_date', 'end_dat
e', 'marital_status', 'rented',
                                                 'family_size', 'no_of_children', 'brand_type', 'category',
'couponValidityDays'], axis=1)
train_target = trainY = trainCampaignCouponTransCustItem.redemption_status
```

In [0]:

```
## Transformation of categorical columns
# Label Encoding:
train_cat_features_ver2 = trainX.apply(LabelEncoder().fit_transform)

#### Finalize X & Y
#temp_1 = np.concatenatetrain_transformed_features = pd.DataFrame(data=train_cat_features_ver2)
train_transformed_features = pd.DataFrame(data=train_cat_features_ver2)

array = train_transformed_features.values
number_of_features = len(array[0])
X = array[:,0:number_of_features]
Y = train_target

# Split into training and validation set
validation_size = 0.2
seed = 7
X_train, X_validation, Y_train, Y_validation = train_test_split(X, Y, test_size=validation_size, random_state
=seed)
#X_train, X_test, y_train, y_test = train_test_split(trainX, trainY, test_size=0.15, random_state=66)
```

In [118]:

```
# Model 2 - RandomForest Classifier
model_rf = RandomForestClassifier()
model_rf.fit(X_train, Y_train)
accuracy_score(Y_validation, model_rf.predict(X_validation))

predict_fn_rf = lambda x: model_rf.predict_proba(x).astype(float)
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of n_
estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [0]:

```
# Line-up the feature names
feature_names = list(train_cat_features_ver2)

# Create the LIME Explainer
explainer = lime.lime_tabular.LimeTabularExplainer(X_train ,feature_names = feature_names,class_names=['No',
'Yes'],
                                                   categorical_features=feature_names,
                                                   categorical_names=feature_names, kernel_width=3)
```

LIME prediction explanation

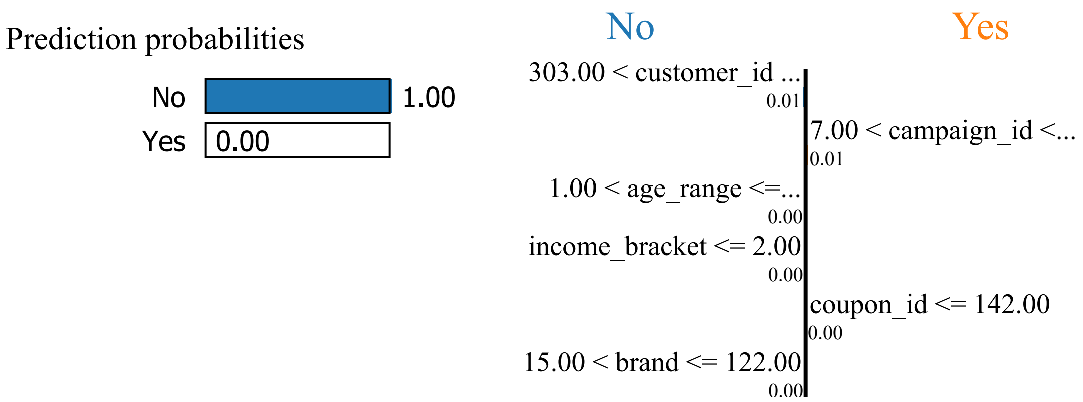
The first plot on the left shows an emphatic prediction of no redemption. The next plot shows that only the campaign id and coupon id are in favor of redemption. The feature weights (shown along the vertical axis) show relatively low weights. Perhaps, more feature engineering will help. The last plot shows the actual values being predicted and explained.

In [120]:

```
# Pick the observation in the validation set for which explanation is required
observation_1 = 2

# Get the explanation for RandomForest
exp = explainer.explain_instance(X_validation[observation_1], predict_fn_rf, num_features=6)
exp.show_in_notebook(show_all=False)
```

```
/usr/local/lib/python3.6/dist-packages/lime/explanation.py:253: ResourceWarning: unclosed file <_io.TextIOWra
pper name='/usr/local/lib/python3.6/dist-packages/lime/bundle.js' mode='r' encoding='utf8'>
  encoding="utf8").read()
```



Feature	Value
customer_id	447.00
campaign_id	9.00
age_range	2.00
income_bracket	0.00
coupon_id	142.00
brand	103.00