# Report: Analysis of Customer Behavior on Purchase of Bikes and Monthly Spend

Fred YANG Runtong, April 2017

## Executive Summary

This document presents findings from data collected from the customers of the company of Adventure Works Cycles. In particular, this report aims at utilizing distinct entries of around 18,000 of the customers' particulars and purchasing histories to establish two models: one that peeks into the customers' inclination of purchasing a bike with their demographical residence, and the other that predicts customers' average monthly spend with the company.

The project is carried out in a streamline manner. Firstly, two raw data source are joined together while ruling out duplicated records based on customer IDs; secondly, data analysis is carried out to view the distributions and relationships among different features; then based on the observations, classification and predictive models are built with benchmarking, feature selection to choose those influential characteristics, and hyper-parameter tuning to optimize the model.

After the analysis, the following conclusions could be drawn:

- Classification model for bike buyers with respect to different geo-locations:
  - A Random Decision Forest model that achieves accuracy of 79.2% with cross validation performed.
  - Most Important 5 features are: Numbers of Cars Owned, Numbers of Children at Home, Yearly Income, Age, and Education.
  - Unfortunately, demographical features are not deterministic in classify customers; in fact, they would bring negative effect to confuse the model.
- Predictive model on customers' monthly spend:
  - A Boosted Decision Tree (for regression) model that achieve MAE (Mean Absolute Error) of 1.498.
  - Top 5 influential features are: Yearly Income, Age, Gender, Marital Status, and Numbers of Children at Home.

## Peek into the Data

There are 2 raw data sources that separately contain customers' personal information (*AWCustomer*: 24 fields, 18,361 rows) and the two attributes this project aims to model (*AWSales*: 3 fields – IDs, Bike Buyer Flag, Average Month Spend, 18,355 rows).

The first observation is that the numbers of entries in two sources have slight disagreement. This implies that at least one source contains duplicates.

Another immediate observation is that the datasets contain many more categorical features versus numerical features, and some seemly numerical attributes are actually categorical in nature (Number of Children at Home, Number of Cars Owned, Total Children). In addition, among 24 fields in customer particulars, at least 7 fields (title, first, middle and last name, suffix, postal code, phone number) contain no significant information for modeling and can be removed, while 2 other fields (Address line 1 & 2) are marginally irrelevant since they are too fine-grained. Lastly, "lastly updated" and "customer ID" would become unnecessary once duplicates are checked.

That being said, besides labels, there are in total 13 fields potentially relevant:

- Locations: City, StateProviceName, CountryRegionName
- Personals: BirthDate, Education, Occupation, Gender, MaritalStatus
- Family: HomeOwnerFlag, NumberChildrenAtHome, TotalChildren, NumberCarOwned, YearlyIncome

## Data Cleansing and Preparation

### Removing Duplicates

By firstly dropping the aforementioned 7 fields, duplicates, identified by repeated appearance of Customer IDs, are solely discovered in *AWCustomer* source, source of errors are:

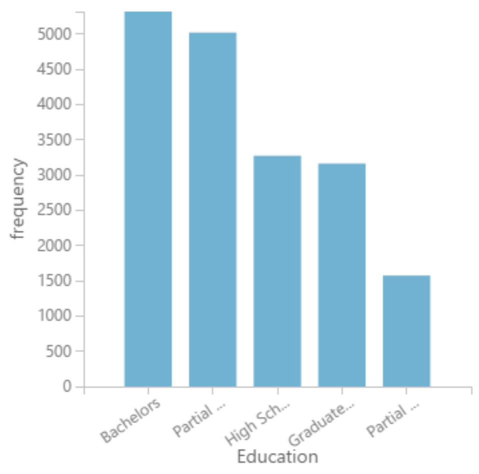
- All attributes are the same (4 occurences)
- Same ID but different Last Updated records (2 occurrences)
- Same ID but different YearlyIncome and Matrital Status (2 occurrences)

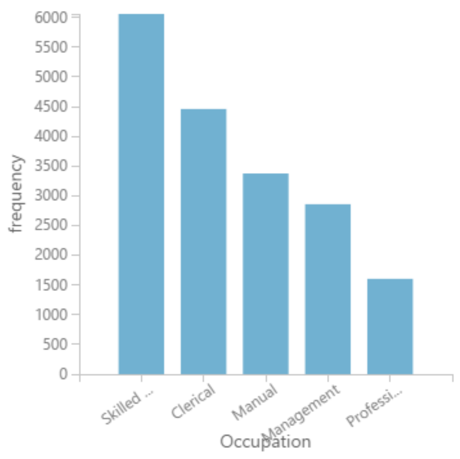
Since duplicates rarely occurred and does not significantly affect the overall distribution of data, only the first occurrence is kept, as a result, both data sources are now of length 18,353.

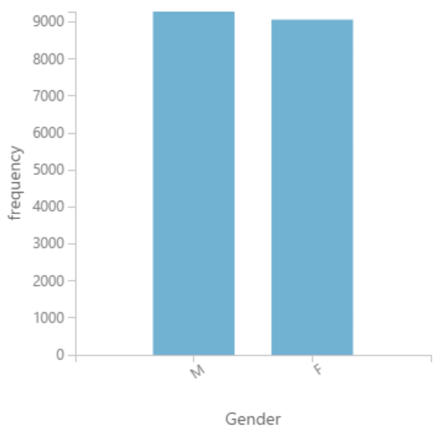
### Statistics and Visualization

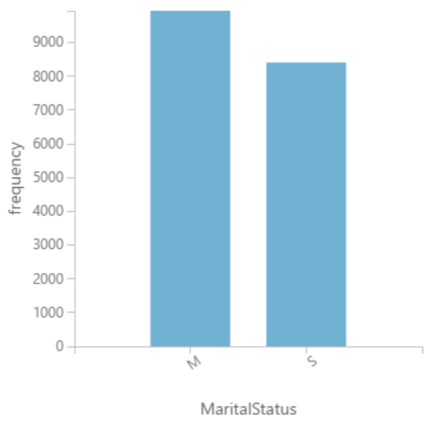
Leveraging visualization capability of Azure ML and Python Matplotlib, attributes categorized as personals and family, except BirthDate (9 fields in total) are presented as follows:

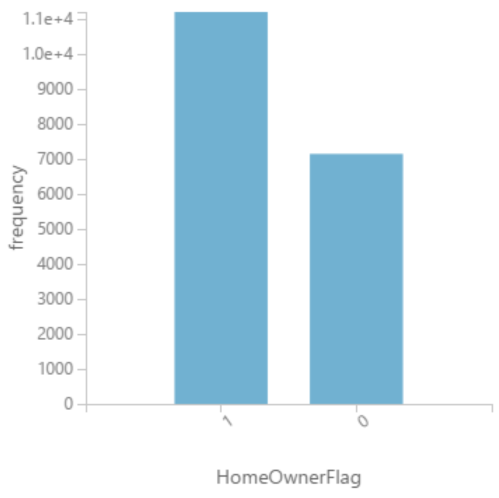
**Categorical Features:**

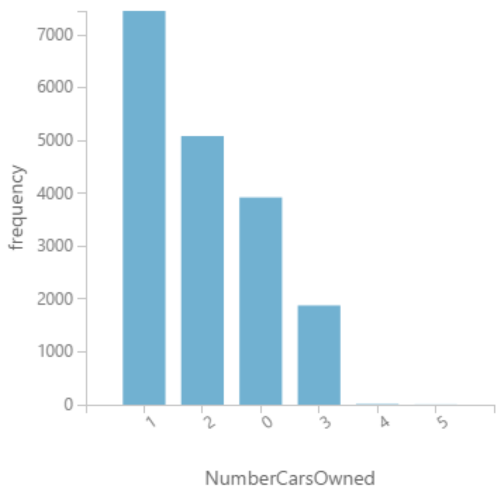


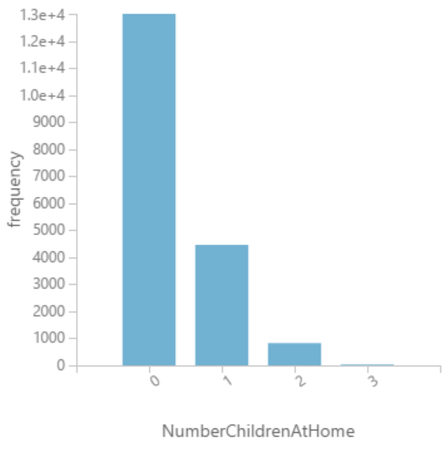


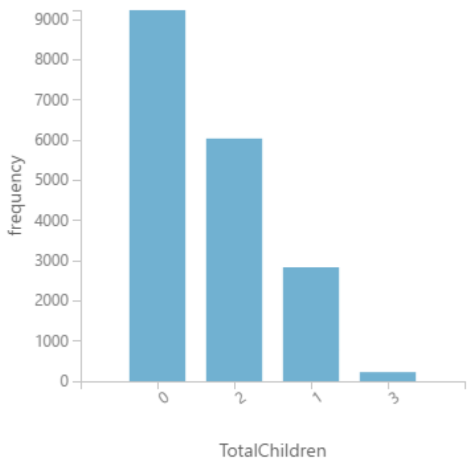








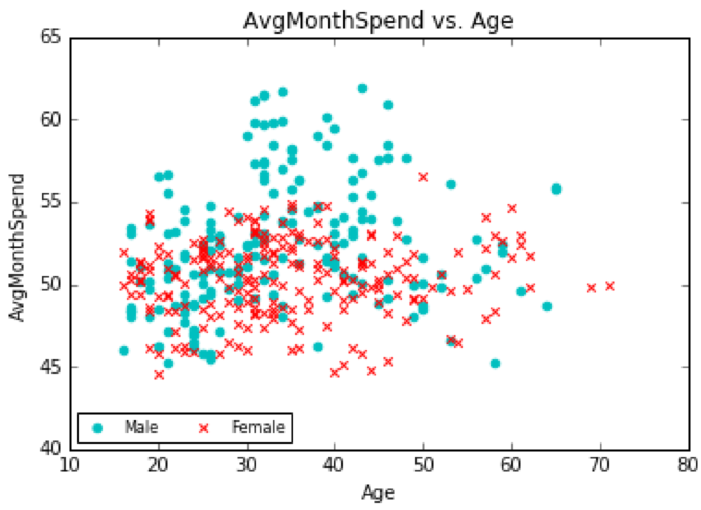




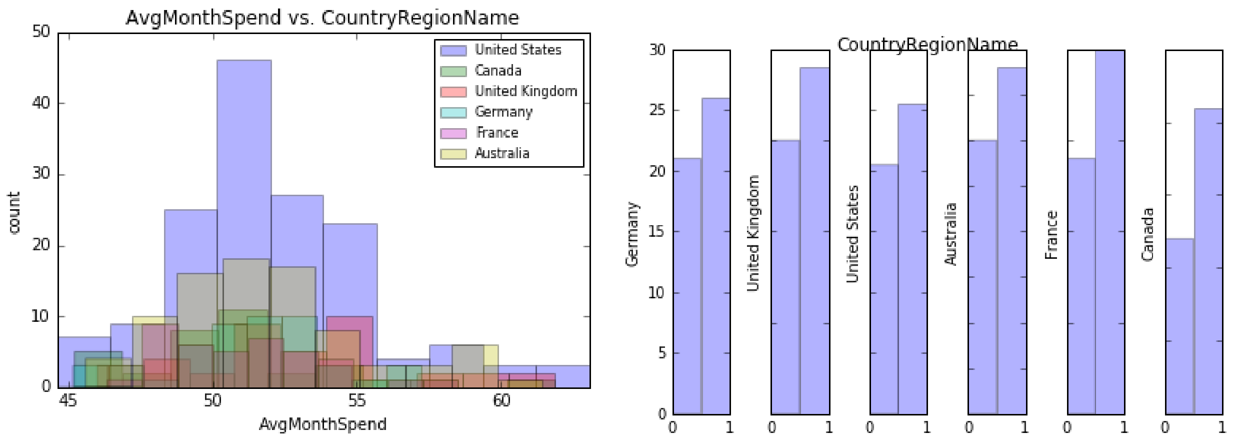
Some Observations:

- <u>Balanced</u> data: Education, Occupation, Gender, MaritalStatus, HomeOwnerFlag. They need no further processing.
- NumberChildrenAtHome, TotalChildren, NumberCarOwned are monotonically <u>decreasing</u>. However, those data are kept as they are (but make categorical) and <u>NOT</u> trimmed to be “0 and more than 0” category, or otherwise some deterministic features might be lost in the transformation.

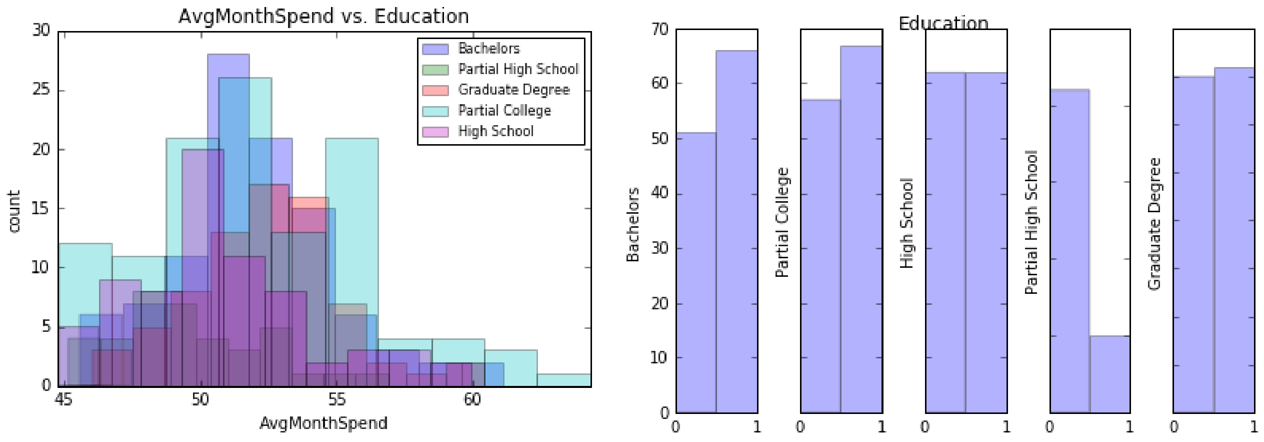
Gearing towards a more targeted comparison with the label feature - average month spend, and bike buyer flags - some distribution tells useful stories (by using a random sample size of 400):



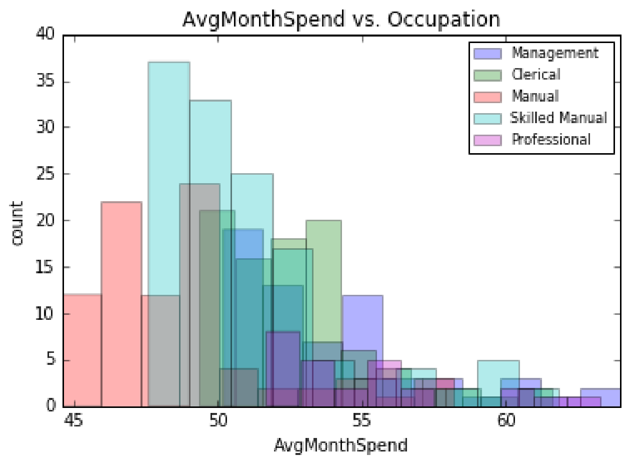
The distribution above tells us that unlike stereotype where young women are heavy spenders, males in their middle ages are likely to purchase the most with the company.

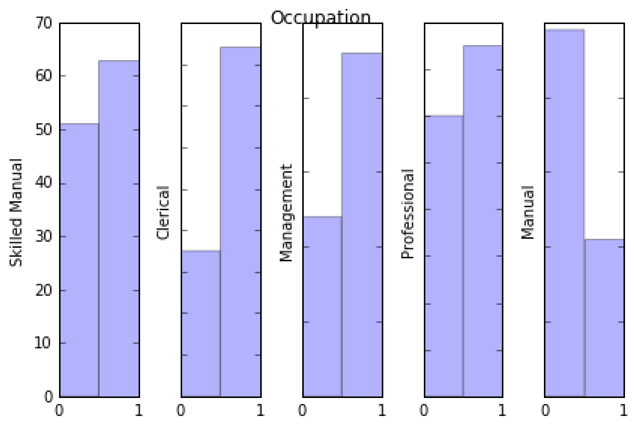


People from the United States surpass other regions in terms of population to a large extent, it might be attribute to the assumption that this company has its most mature business rooted in the U.S. In the category of bike-buyer though, customers' behavior does not vary much across different regions.

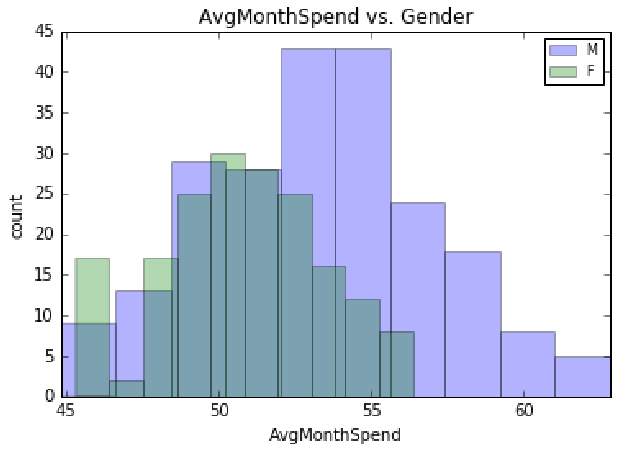


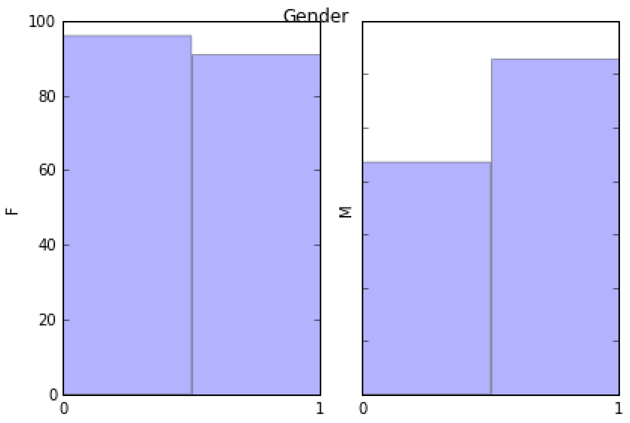
People’s monthly spend does not seem to vary across different educational levels, but it indeed show that certain group are much less likely to buy a bike (e.g. Partial high School)



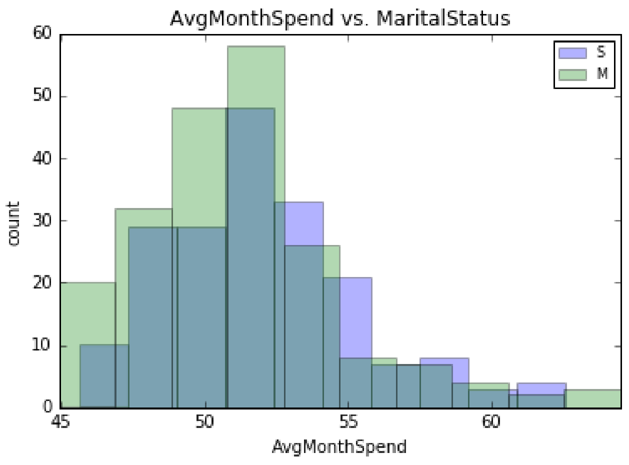


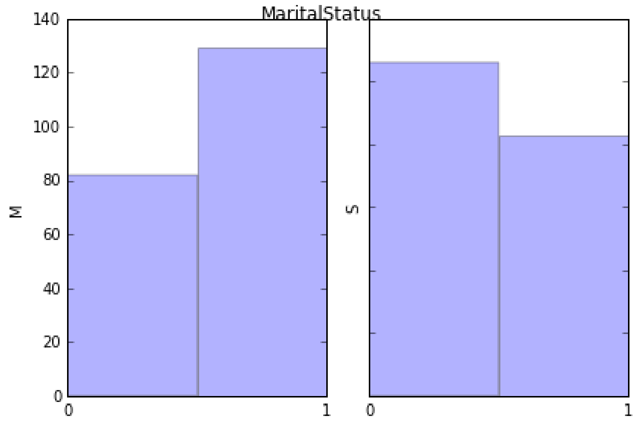
Categorized by occupation, it can be deduced that people’s buying habits are predominantly affected by their positions, which could be an indicator of their earning capability. Manual-ranking persons spend obviously less than those in the management positions, and they do vary in terms of their inclination of buying a bike also.



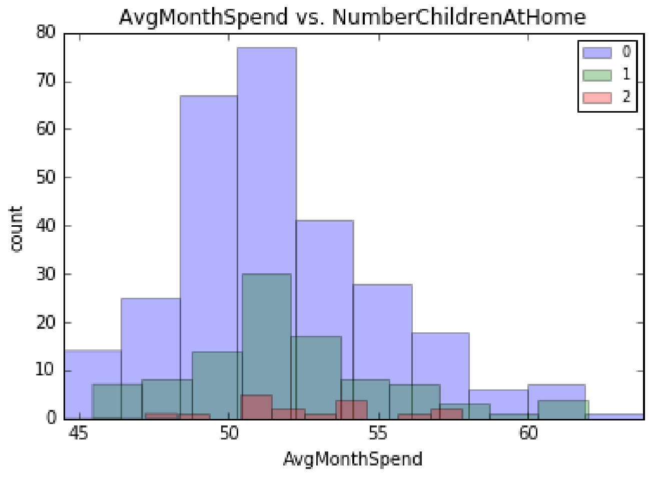


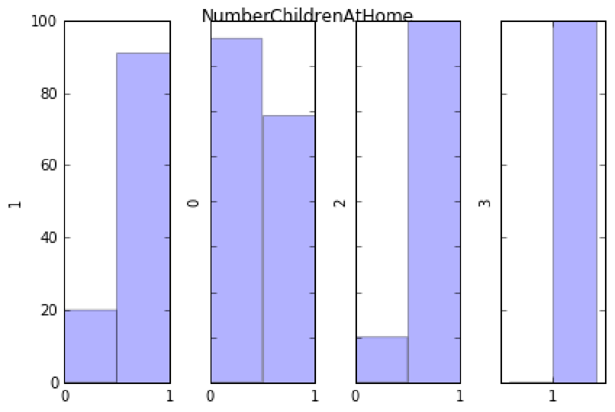
Both group shares the same distribution, though male customers have a larger range, and this is mostly contributed by their heavy spenders (average monthly spend > $55). Additionally, a male customer is more likely to be a bike rider.

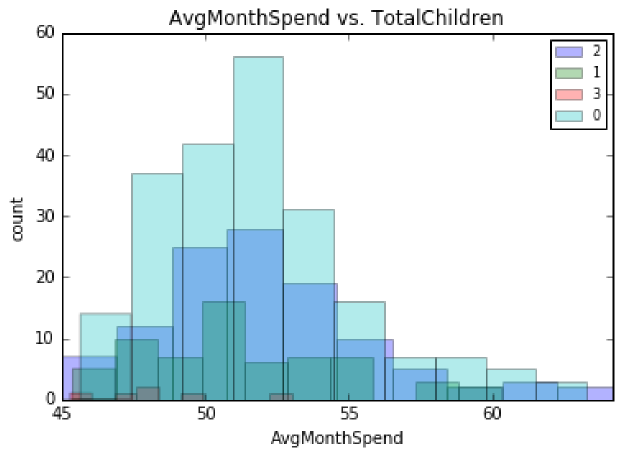


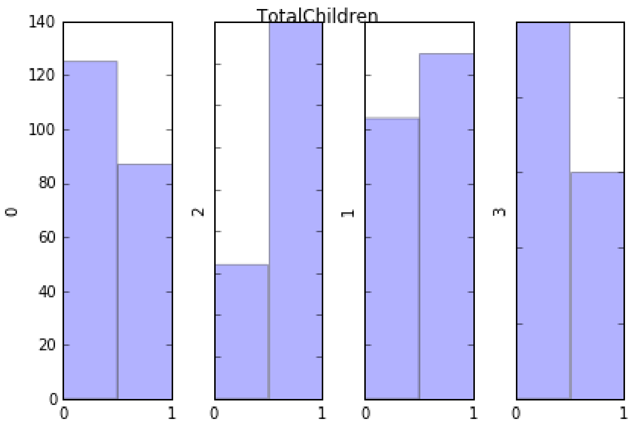


People’s purchasing does not seem to be affected by their marital status, but single persons are less inclined to buy bikes nevertheless.



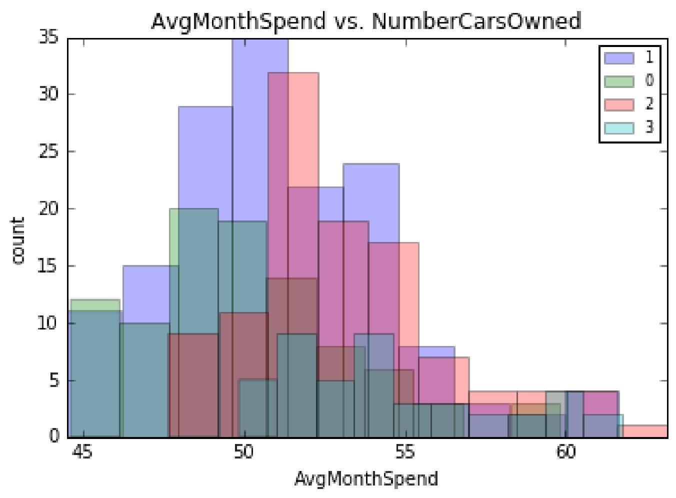


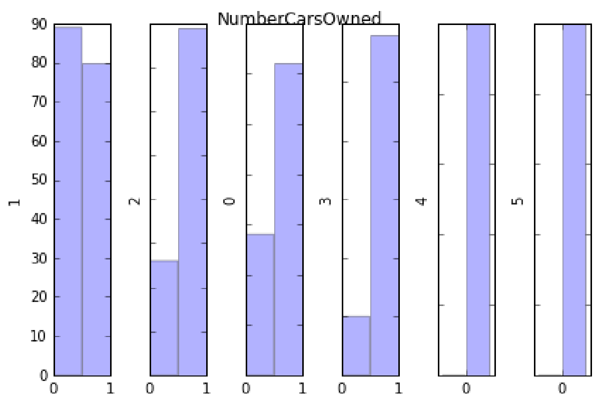




Though monthly spend does not vary across families of different children (though cannot tell from those with more than 2 children since sample size is too small), they do affect the family’s decision whether to own a bike. Especially with more children AT HOME, families are much more likely to purchase a bike (probably as a Christmas gift for their children).

In addition, though on the surface two parameters seems to correlates with each other (and indeed in terms of their correlation with monthly spend), they do vary in the region of determining bike buyers. For example, the abnormal fact comes when total children increased to 3 in a family, the increasing trend is suddenly reversed. Hence, whether to rule out one of the two variables would be decided in the later stage (i.e. perturbation-based feature importance).





Lastly, with more cars at home, one customer is more likely to be a heavy spender with the company. And there is also a positive correlation with bike-buyer indicator, though the trend does not hold when numbers of cars owned varying from 0 to 1. One possible cause would be that if a family does not own a car at all (especially in America), it is very likely that the family would live in a thrifty substantiation to the extent that whether to purchase a bike becomes a careful decision for the person.

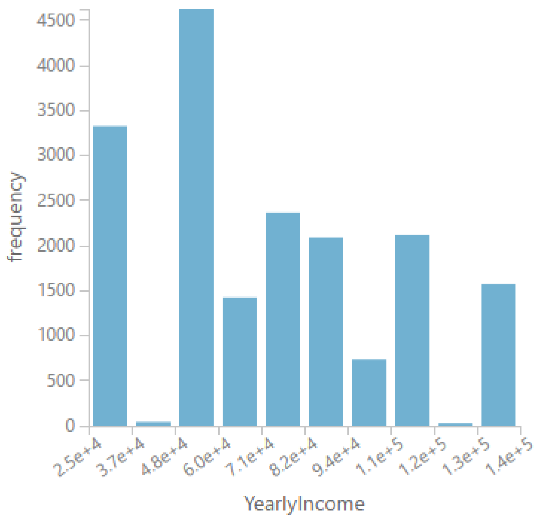
With the visualization before, some factors' assumed importance are listed as follows:

(√: important, ×: not important, ?: cannot tell)

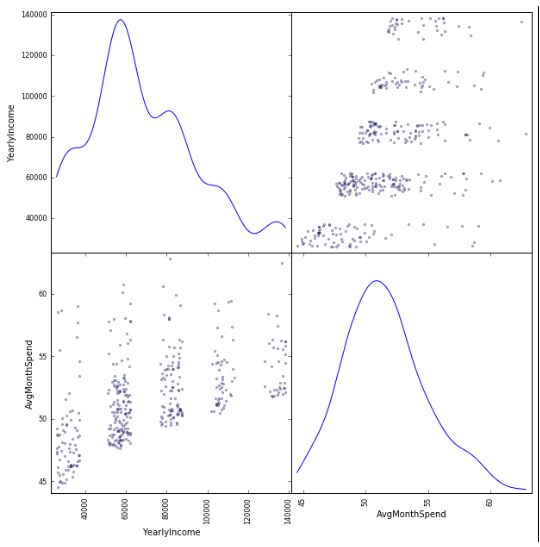
| *Features* | AvgMonthSpend | BikeBuyer |
|---|---|---|
| *Age* | √ | ? |
| *CountryRegionName* | × | × |
| *Education* | √ | √ |
| *Occupation* | √ | √ |
| *Gender* | √ | √ |
| *MaritalStatus* | × | √ |
| *NumberChildrenAtHome* | ? | √ |
| *NumberTotalChildren* | ? | √ |
| *NumberCarOwned* | √ | √ |

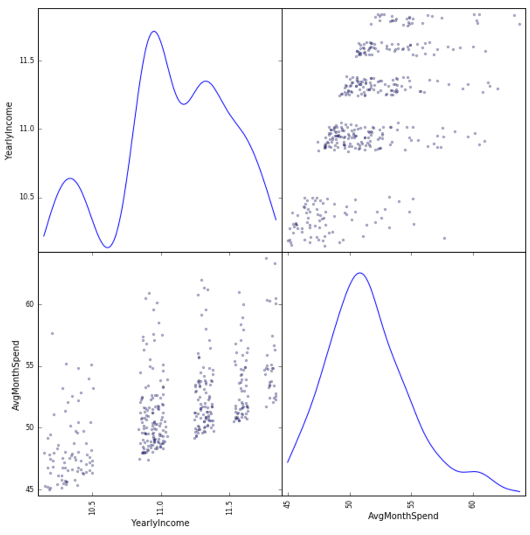
## Numerical Features

The only numerical feature left is Yearly Income.



It does not fall into normal distribution, but a comparison with our targeting label, the other numerical feature: average monthly spend, gives a clearer picture of the distribution.





Original Scale Scatter Matrix                                        Log Scale Scatter Matrix

Though many points fall into the long tail, therefore outlier region, the dense area clearly shows a positive relationship between AvgMonthSpend and YearlyIncome. However, the trending is not linear, this implies a <u>log scale transformation</u> might be needed in the subsequent treatment.

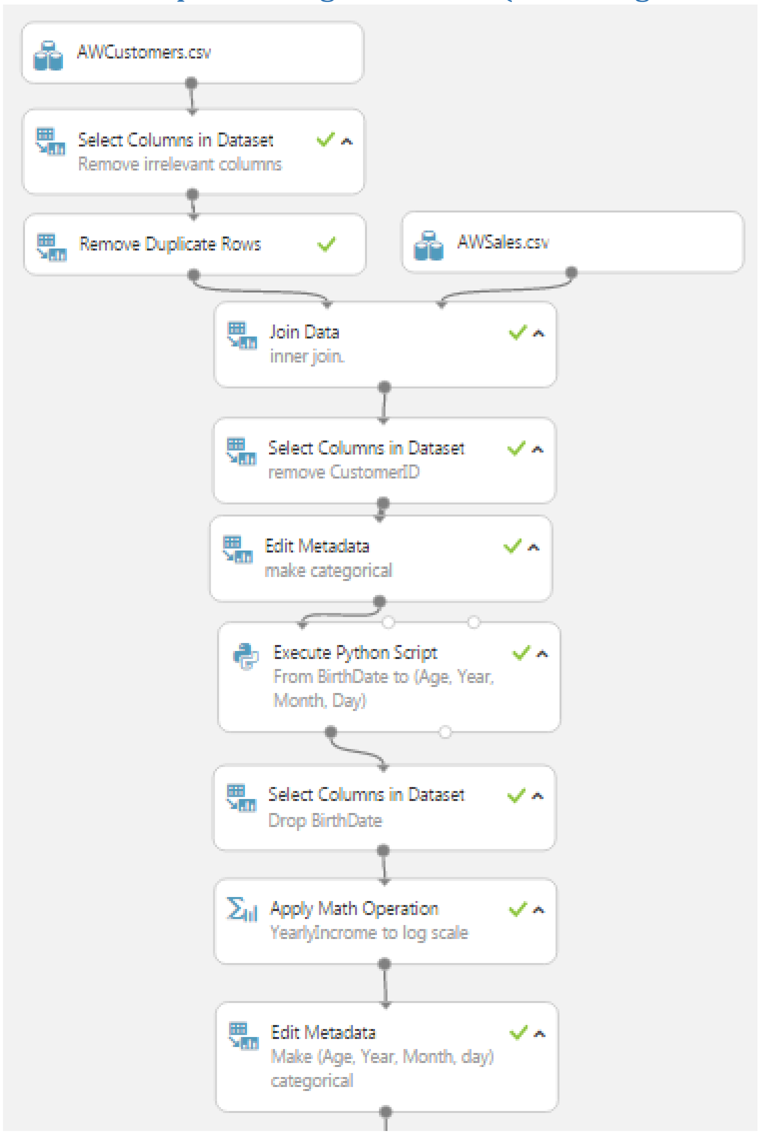
## Data Wrangling and Transformation

### A Short Summary of Considerations

By exploring data above, some decided data wrangling process would be:

- Make every input, except *YearlyIncome*, categorical features
- Segment *BirthDate* to Year, Month, Day, and calculate Age (later experiment shows that year/month/day are not proper features, dropped thereafter)
- Change *YearlyIncome* to log scale

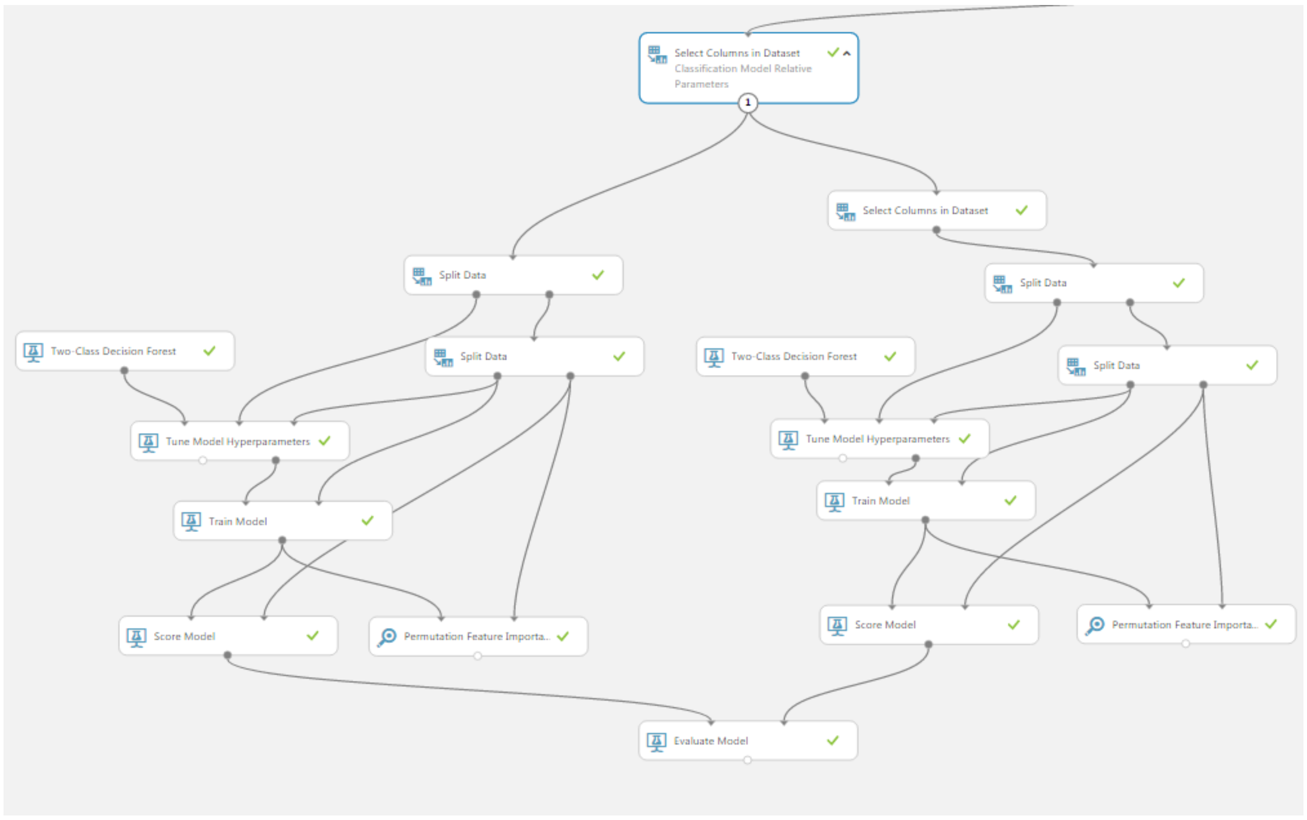
### An Overview of Azure ML Pre-processing Workflow (including cleansing and wrangling)


AWCustomers.csv
Select Columns in Dataset
Remove irrelevant columns
Remove Duplicate Rows
AWSales.csv
Join Data
inner join.
Select Columns in Dataset
remove CustomerID
Edit Metadata
make categorical
Execute Python Script
From BirthDate to (Age, Year, Month, Day)
Select Columns in Dataset
Drop BirthDate
Apply Math Operation
YearlyIncrome to log scale
Edit Metadata
Make (Age, Year, Month, day) categorical

## Feature Selections and Modelling

### Classification Model for Bike Buyer

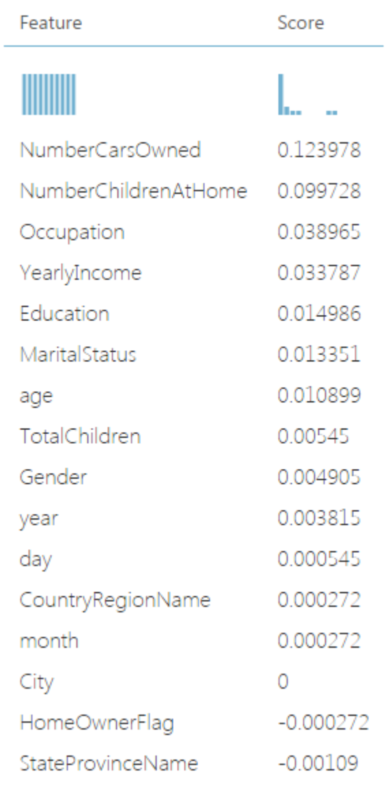
Removing Average Monthly Spend out of the dataset and make Bike Buyer as the score label, the left branch of initial model is built. Decision Forest is used as the kernel for the classifier, and hyper-parameters are tuned for optimized result.


Select Columns in Dataset
Classification Model Relative Parameters
Select Columns in Dataset
Split Data
Split Data
Two-Class Decision Forest
Split Data
Two-Class Decision Forest
Split Data
Tune Model Hyperparameters
Tune Model Hyperparameters
Train Model
Train Model
Score Model
Permutation Feature Importa...
Score Model
Permutation Feature Importa...
Evaluate Model

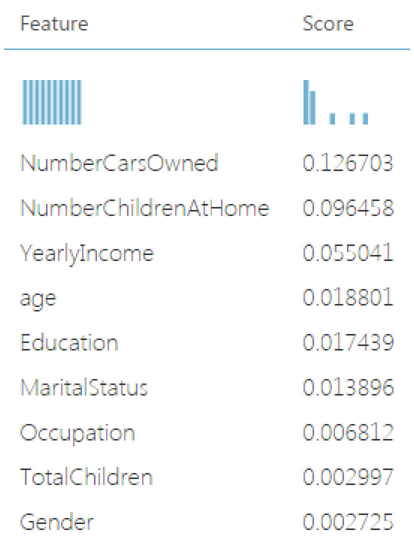
From the feature importance though, it could be observed that some features are irrelevant or harming the accuracy of the model:

- Generated features: Year (year is not irrelevant viewing from score, but it is actually the repeated version of age), Month, Day
- Geolocation features: City, StateProvinceName
- HomeOwnerFlag

Removing those features and the right branch is showed to contain less attributes.



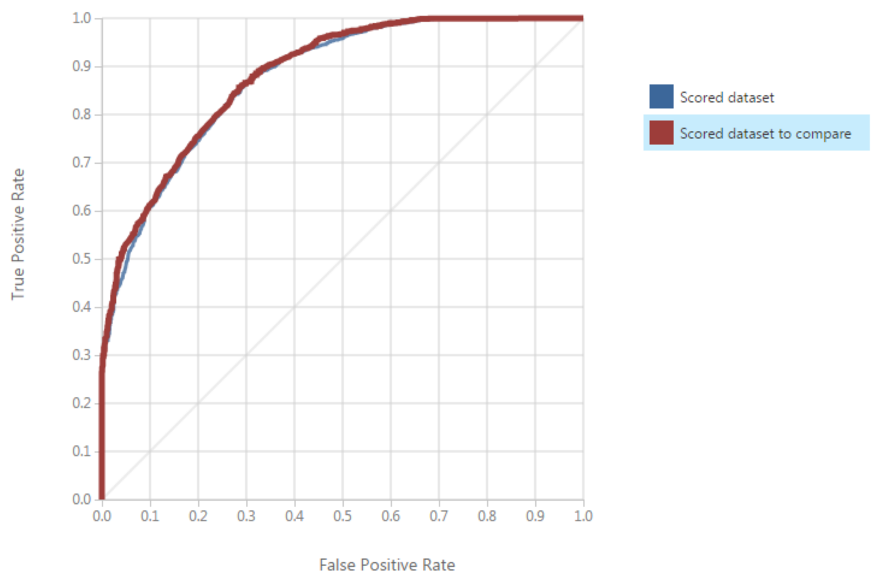
| Feature | Score |
|---|---|
| NumberCarsOwned | 0.123978 |
| NumberChildrenAtHome | 0.099728 |
| Occupation | 0.038965 |
| YearlyIncome | 0.033787 |
| Education | 0.014986 |
| MaritalStatus | 0.013351 |
| age | 0.010899 |
| TotalChildren | 0.00545 |
| Gender | 0.004905 |
| year | 0.003815 |
| day | 0.000545 |
| CountryRegionName | 0.000272 |
| month | 0.000272 |
| City | 0 |
| HomeOwnerFlag | -0.000272 |
| StateProvinceName | -0.00109 |



| Feature | Score |
|---|---|
| NumberCarsOwned | 0.126703 |
| NumberChildrenAtHome | 0.096458 |
| YearlyIncome | 0.055041 |
| age | 0.018801 |
| Education | 0.017439 |
| MaritalStatus | 0.013896 |
| Occupation | 0.006812 |
| TotalChildren | 0.002997 |
| Gender | 0.002725 |

left branch feature importance right branch feature importance

Evaluation ROC also reveals that even with great extent of feature pruning, accuracy is not affected.

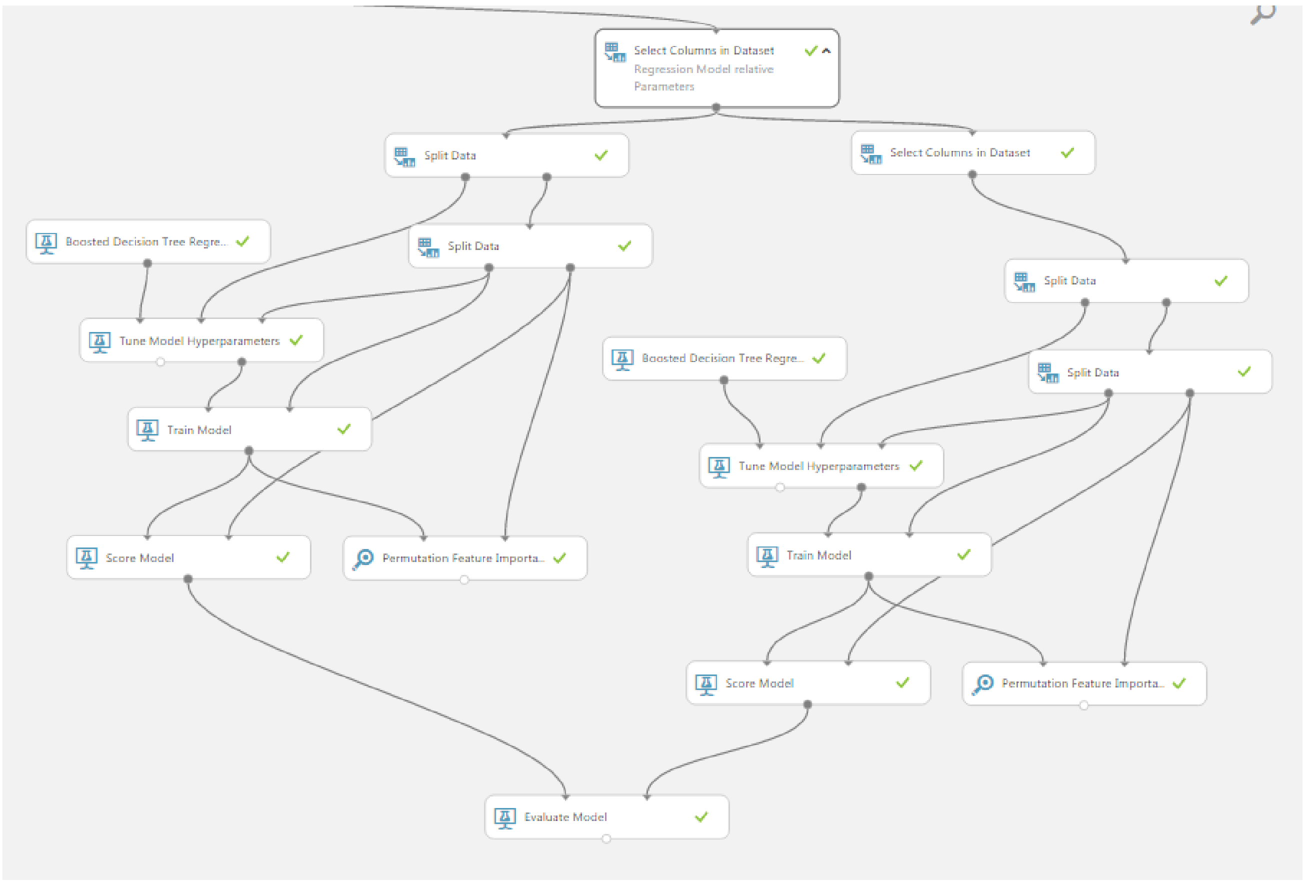


The associated stats are:

| Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|
| **0.794** | 0.799 | 0.883 | 0.827 |

## Regression Model for Average Monthly Spend

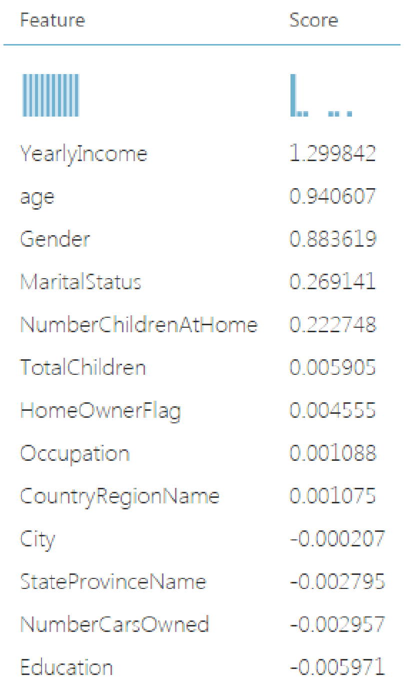
Removing Bike Buyer out of the dataset and make Average Monthly Spend as the score label, the left branch of initial model is built. Boosted Decision Tree is used as the kernel for the classifier, and hyper-parameters are tuned for optimized result.


Select Columns in Dataset
Regression Model relative Parameters
Split Data
Select Columns in Dataset
Boosted Decision Tree Regre...
Split Data
Split Data
Tune Model Hyperparameters
Boosted Decision Tree Regre...
Split Data
Train Model
Tune Model Hyperparameters
Score Model
Permutation Feature Importa...
Train Model
Score Model
Permutation Feature Importa...
Evaluate Model

From the feature importance though, it could be observed that some features are irrelevant or harming the accuracy of the model:

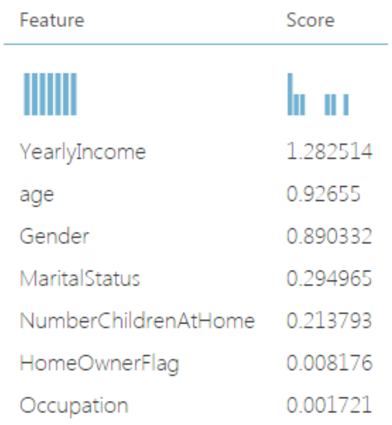
- Education
- NumberCarsOwned
- City, StateProvinceName

Removing those features and the right branch is showed to contain less attributes.



| Feature | Score |
|---|---|
| YearlyIncome | 1.299842 |
| age | 0.940607 |
| Gender | 0.883619 |
| MaritalStatus | 0.269141 |
| NumberChildrenAtHome | 0.222748 |
| TotalChildren | 0.005905 |
| HomeOwnerFlag | 0.004555 |
| Occupation | 0.001088 |
| CountryRegionName | 0.001075 |
| City | -0.000207 |
| StateProvinceName | -0.002795 |
| NumberCarsOwned | -0.002957 |
| Education | -0.005971 |

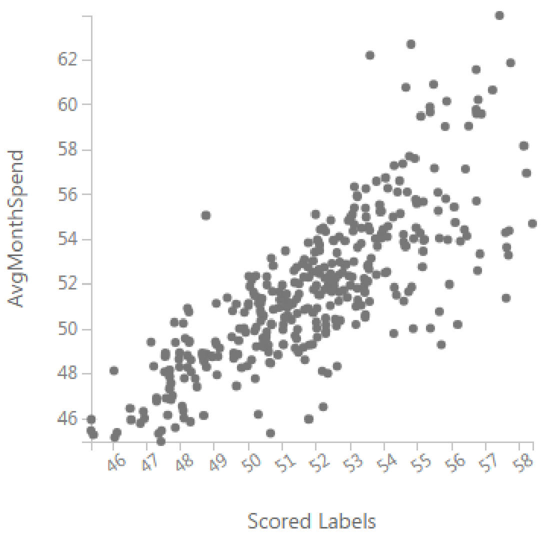
left branch feature importance



| Feature | Score |
|---|---|
| YearlyIncome | 1.282514 |
| age | 0.92655 |
| Gender | 0.890332 |
| MaritalStatus | 0.294965 |
| NumberChildrenAtHome | 0.213793 |
| HomeOwnerFlag | 0.008176 |
| Occupation | 0.001721 |

right branch feature importance

With a sample rate of 0.1, the comparison between actual and estimated values for test data clearly shows a linear relationship, albeit the performance is less satisfying at the high side.



The associated stats are:

| | |
|---|---|
| **Mean Absolute Error** | **1.496859** |
| **Root Mean Squared Error** | **2.031892** |

Regression model is also built on Python Platform. Code could be view in the Appendix.

## Conclusion

This analysis has demonstrated two models that separately predict the average monthly spend of a customer from his personal particulars and past purchasing history, and the possibility that he could become a potential purchaser of bikes. In particular, the analysis also shows that regional variance does not play any significant roles in neither of the models, which indicates that demographical distribution of customers are not as important as their personal statuses. In addition, different models absorb different sets of parameters to evaluate. While Numbers of Cars Owned, Numbers of Children at Home, and Yearly Income are the most important features for the classifier, Yearly Income, Age, and Gender becomes more important when determining the customers' monthly spend.

# Appendix

## Sample Python Code for Visualization

```python
# Histogram on different categorical values
def hist_AvgMonthSpend_plot(items):
    df_sample = df.sample(n=400)
    import matplotlib.pyplot as plt
    category=df_sample[items].unique()
    for entry in category:
        plt.hist(df_sample[df_sample[items]==entry].AvgMonthSpend, bins=10, alpha=0.3)
    plt.title("AvgMonthSpend vs. %s"%items)
    plt.xlabel("AvgMonthSpend")
    plt.ylabel("count")
    plt.legend(category, fontsize=8)
    plt.xlim([df_sample['AvgMonthSpend'].min(), df_sample['AvgMonthSpend'].max()])
    plt.show()


def hist_BikeBuyer_plot(items):
    df_sample = df.sample(n=400)
    import matplotlib.pyplot as plt
    import matplotlib.ticker as ticker
    columns = df_sample[items].unique()
    f, axarr = plt.subplots(1, len(columns))
    for num in range(0,len(columns)):
        axarr[num].hist(df_sample[df_sample[items]==columns[num]].BikeBuyer, bins=2, alpha=0.3)
        if num != 0:
            plt.setp(axarr[num].get_yticklabels(), visible=False)
        axarr[num].xaxis.set_major_locator(ticker.MultipleLocator(1.0))
        axarr[num].yaxis.set_label_position("left")
        axarr[num].set_ylabel(columns[num])
    f.subplots_adjust(hspace=0.5)
    f.suptitle(items, fontsize=12)
    plt.tight_layout()
    plt.show()


hist_plot_list = ['CountryRegionName','Education','Occupation','Gender','MaritalStatus',
                  'NumberChildrenAtHome','TotalChildren', 'NumberCarsOwned']
```

```python
for name in hist_plot_list:
    hist_AvgMonthSpend_plot(name)
    hist_BikeBuyer_plot(name)
```

## Sample Python Code for Sklearn Regression Models

```python
import pandas as pd
# these data are all processed data as described in the former sections
train_path = "Regressiontrain-data-processed.csv"
test_path = "Regressiontest-data-processed.csv"
test_ID_path = "AWTest-Regression.csv"

def encoding(df):
    # data wrangling: encoding categorical/string values
    from sklearn import preprocessing
    string_val_list = [
            'Occupation',
            'Gender',
            'MaritalStatus',
            'CountryRegionName'
            ]
    for items in string_val_list:
        le = preprocessing.LabelEncoder()
        le.fit(df[items].unique())
        df[items] = le.transform(df[items])
    return df

train_df = encoding(pd.read_csv(train_path, index_col=0))
test_df = encoding(pd.read_csv(test_path, index_col=0))
CustomerID = pd.read_csv(test_ID_path).CustomerID

print train_df.dtypes
print test_df.dtypes

import numpy as np
train_df.YearlyIncome = np.log(train_df.YearlyIncome)
test_df.YearlyIncome = np.log(test_df.YearlyIncome)

# train classification model
y = train_df['AvgMonthSpend'].copy()
```

```
X = train_df.drop(labels=['AvgMonthSpend'], axis=1)
from sklearn import cross_validation
data_train, data_test, label_train, label_test = cross_validation.train_test_split(X, y, test_size=0.33, random_state=1)

from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(max_depth=8, random_state=1)
model.fit(X,y)
predict_decisionTree = pd.DataFrame({"CustomerID":CustomerID,"predict-tree":model.predict(test_df)})

from sklearn import linear_model
model = linear_model.LinearRegression()
model.fit(X,y)
predict_decisionForest = pd.DataFrame({"CustomerID":CustomerID,"predict-linear":model.predict(test_df)})

joined_label = predict_decisionTree.set_index('CustomerID').join(predict_decisionForest.set_index('CustomerID'))
joined_label['diff'] = joined_label['predict-tree'] - joined_label['predict-linear']
joined_label['avg'] = (joined_label['predict-tree'] + joined_label['predict-linear'])/2
```