Outlet Sales Forecasting with Machine Learning

Outline

01 Exploratory Data Analysis

02 Data Preprocessing & Feature Engineering

03 Model Building & Validation

04 Prediction & Future Improvements

EDA

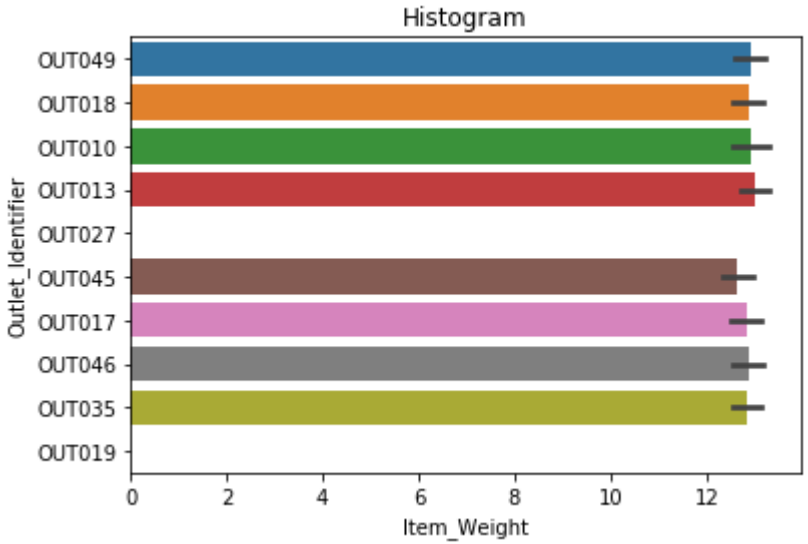
What is the target variable?

And what are the features we have?

- From the descriptive analysis we could find out the distribution of target variables is not normalized.
- We also found out that there are both continuous and categorical missing values and duplicated dimensions existed in our features.

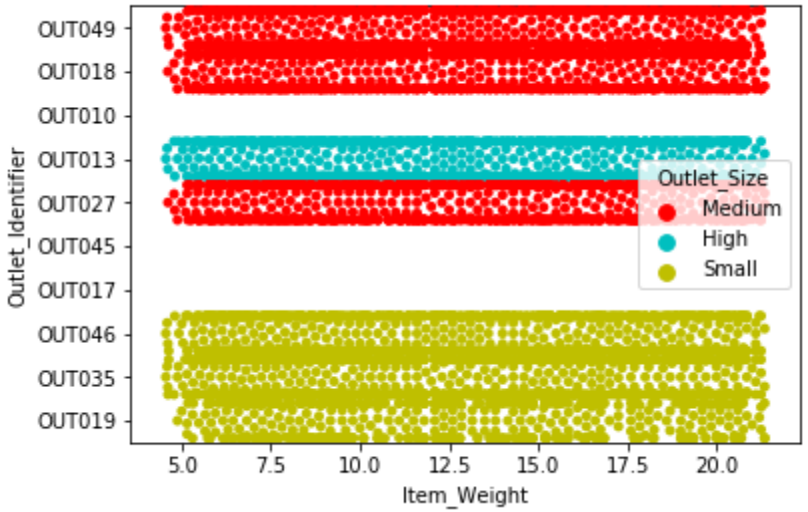
Data Preprocessing

From the chart, we could see that the Item weight is not missing at random but limited to two Outlet ids.



Data imputation would be conducted based on
The findings, we randomly chose item weights from
Those two kinds of outlet to fill in as the actual item weight
For the missing values.

Data Preprocessing



From the scatterplot, we could see that the feature Outlet Size is not missing at random but only happened with three specific Outlet ids.

Missing values for the categorical variables would be imputed based on The specific outlet ids and its corresponding Outlet Size and Item sales. For example, We find out that Out010 corresponds to grocery store therefore The missing values for Outlet Size should be small.

Feature Engineering

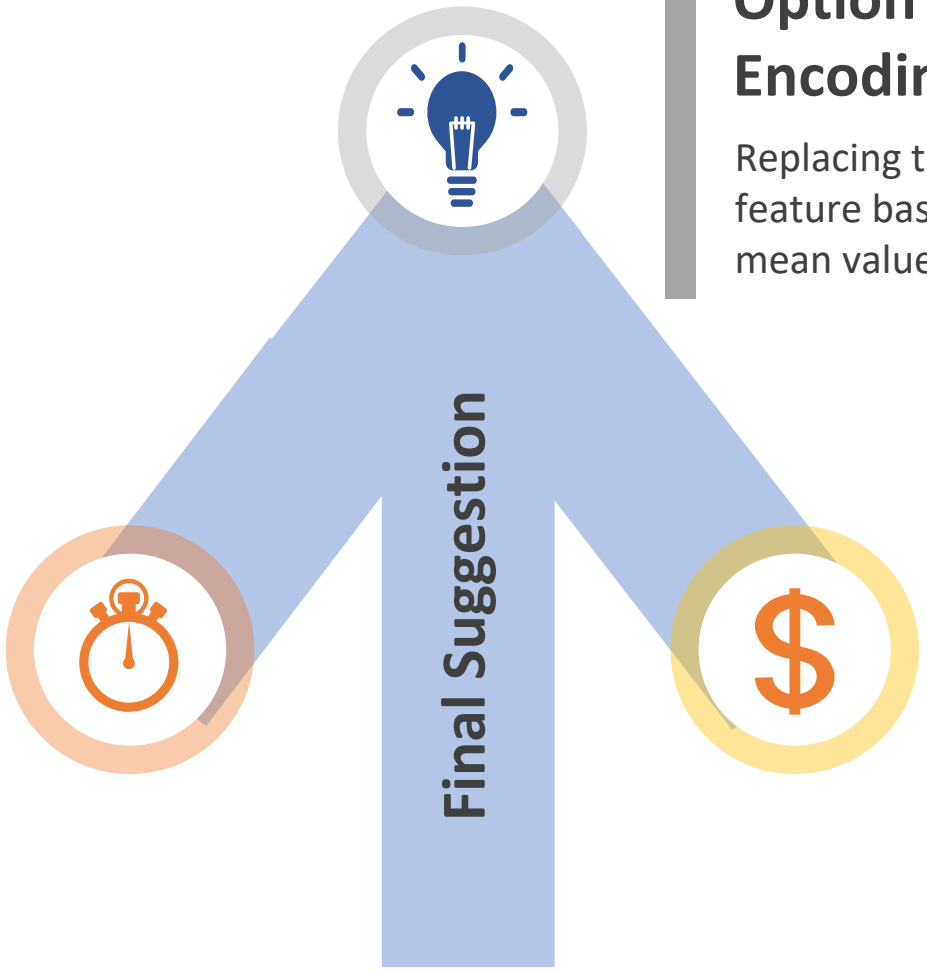
Encoding

Option A(Target Encoding)

Replacing the categorical feature based on the target mean value of each category.

Option B (Label-Encoding)

Convert categorical features into numerical features based on the feature's ordinary sequence

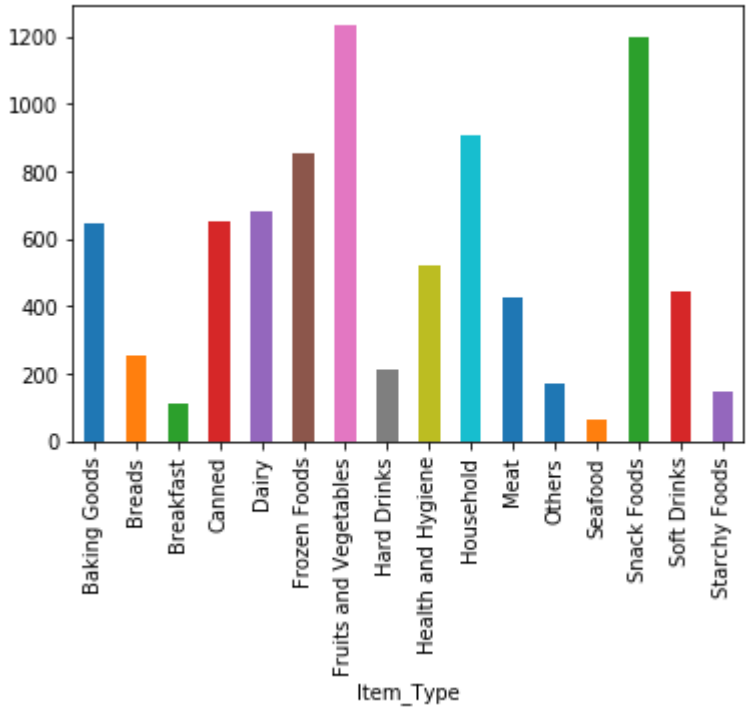


Option C(One-Hot Encoding)

Encode each categorical feature by creating a new binary column.

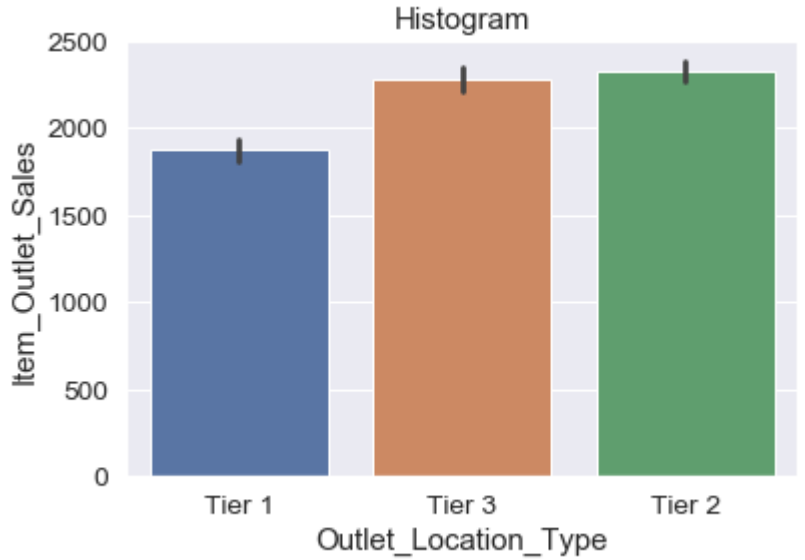
Selling Level

Different Item type differs in their sales figures.



Location Level

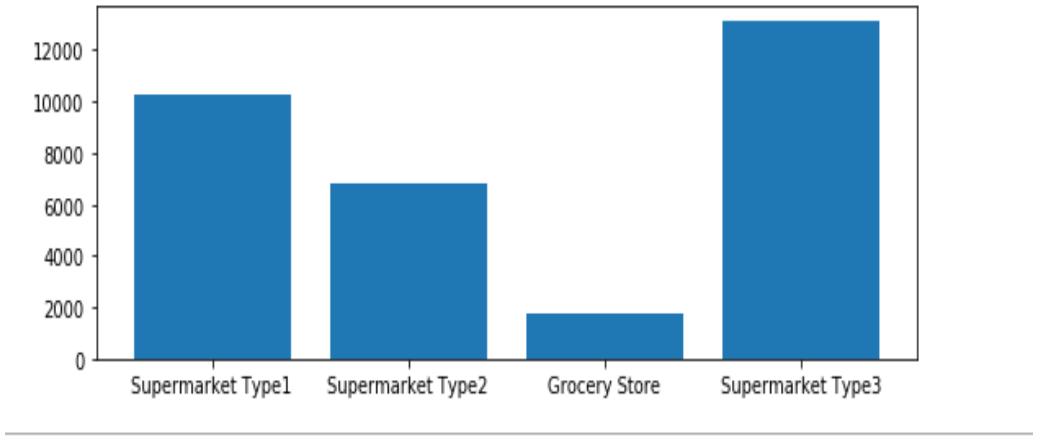
Location matters to the outlet as well as to the sales.



Feature Engineering

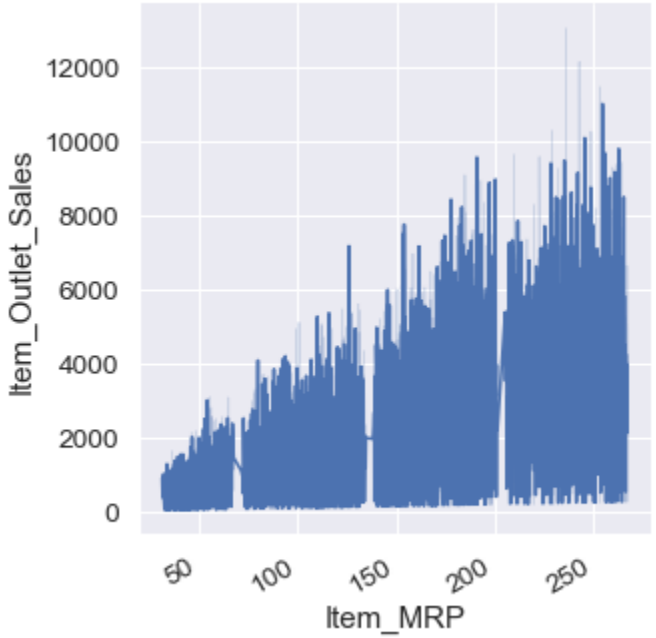
Outlet level

Whether an outlet is a grocery store or not.



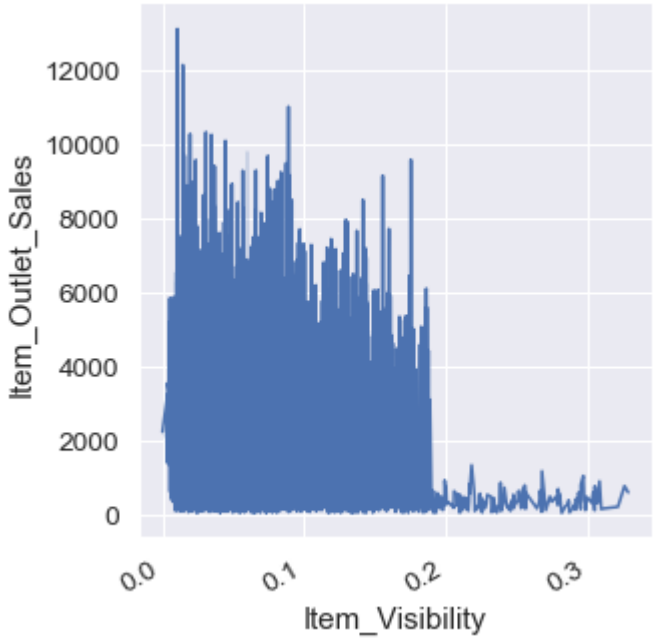
MRP Level

Different level of MRP corresponds to different level of sales.



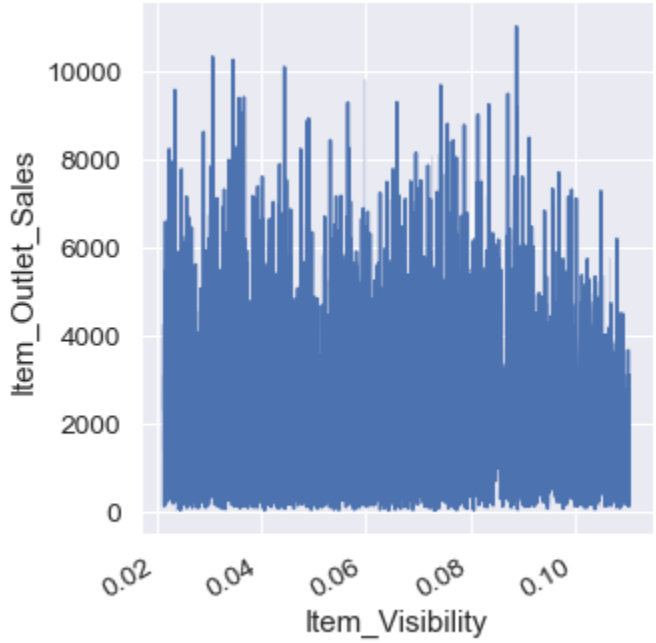
Item Visibility

There is a group of outliers in the item visibility feature, we decide to apply winsorization to them. Limit those extreme values to reduce the effects of spurious outliers.

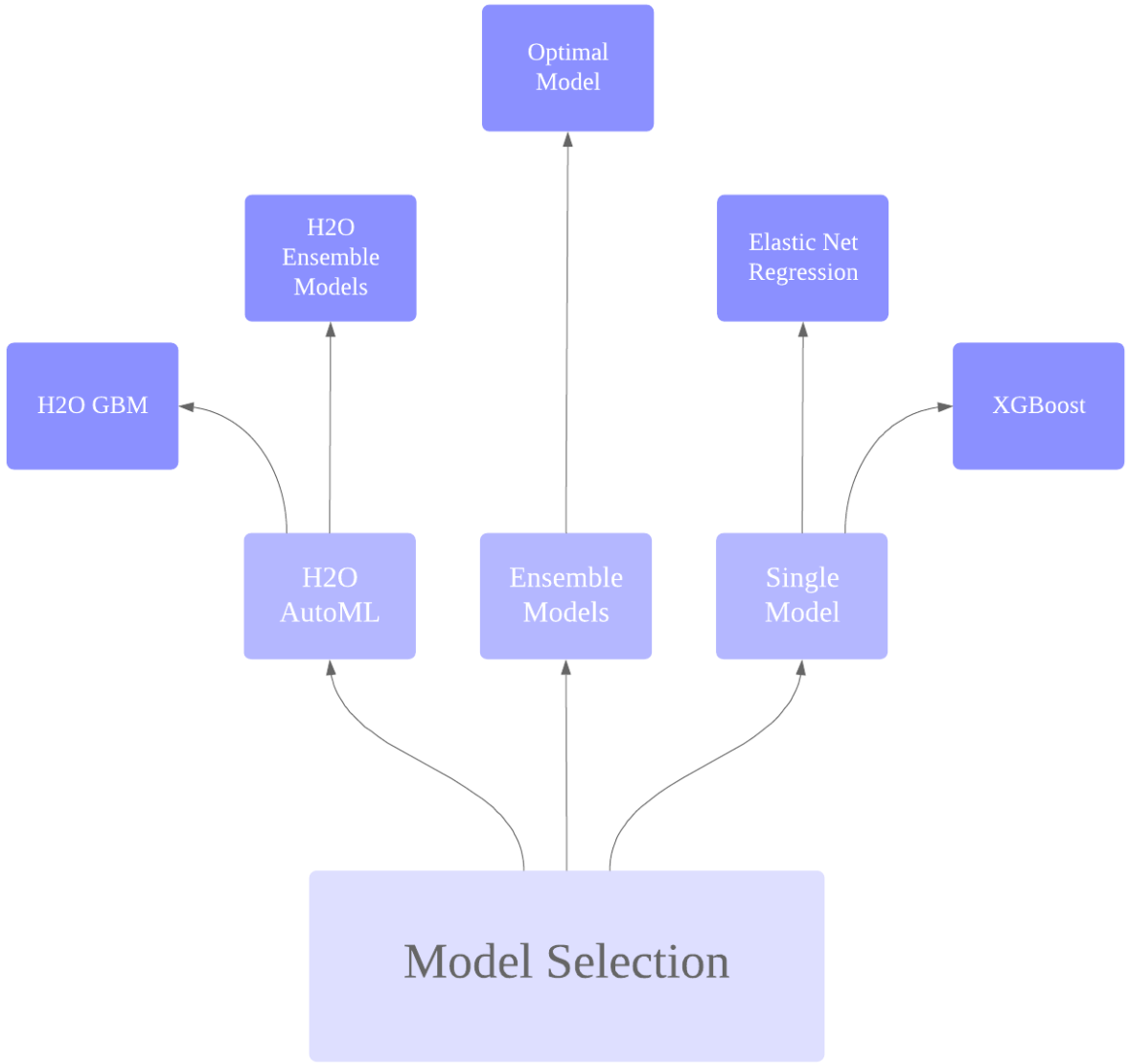


After Winsorization

The extreme values has successfully been transformed within the range and the feature looks fine.



Model Building Iteration



Optimal Model

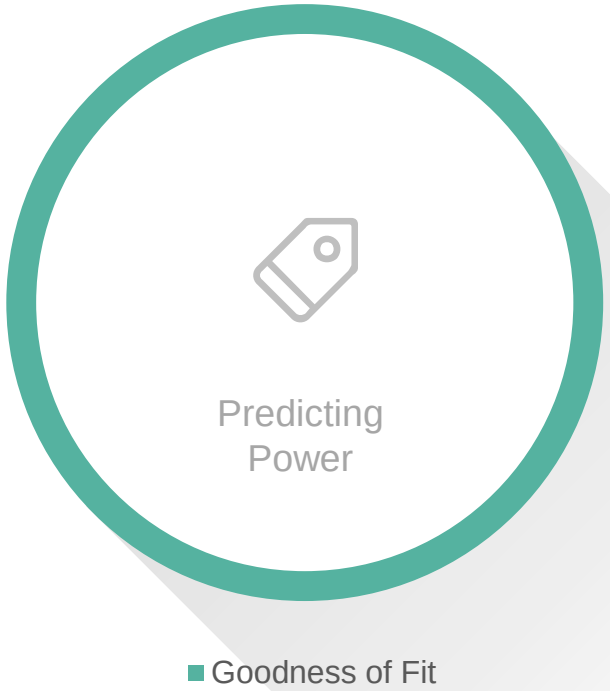
```
lr = LinearRegression()
svr_lin = SVR(kernel='linear')
ridge = Ridge(random_state=1)
lasso = Lasso(random_state=1)
reg = xgb.XGBRegressor()
svr_rbf = SVR(kernel='rbf')
regressors = [svr_lin, lr, ridge,lasso,reg]
stregr = StackingRegressor(regressors=regressors,
                           meta_regressor=svr_rbf)

params = {'lasso__alpha': [0.1, 0.5, 1.0],
          'ridge__alpha': [0.1, 0.5, 1.0],
          'svr__C': [0.1, 0.5, 1.0],
          #'reg__max_depth':[3,4,5],
          #'reg__n_estimators':[10,20,30],
          #'reg__learning_rate':[0.1,0.3,0.5],
           'meta-svr__C': [0.1, 1.0, 5.0],
           'meta-svr__gamma': [0.1, 1.0, 5.0]}

bestmodel4 = RandomizedSearchCV(estimator=stregr,
                                param_distributions=params,
                                cv=5,
                                n_iter=10,n_jobs=-1,random_state=1,scoring='r2')
bestmodel4.fit(train_x, train_y1)
```

Notebook Display

Model Validation



75.65 %

We used cross-validation to evaluate our model since test.csv did not provide the label. Our model finally achieved a **75.65%** goodness of fit score(R^2) with an RMSE of 1.76. Although it is not our ultimate performance since parameter tuning is not finished due to time limit and complexity, it still indicated that our model performed well.

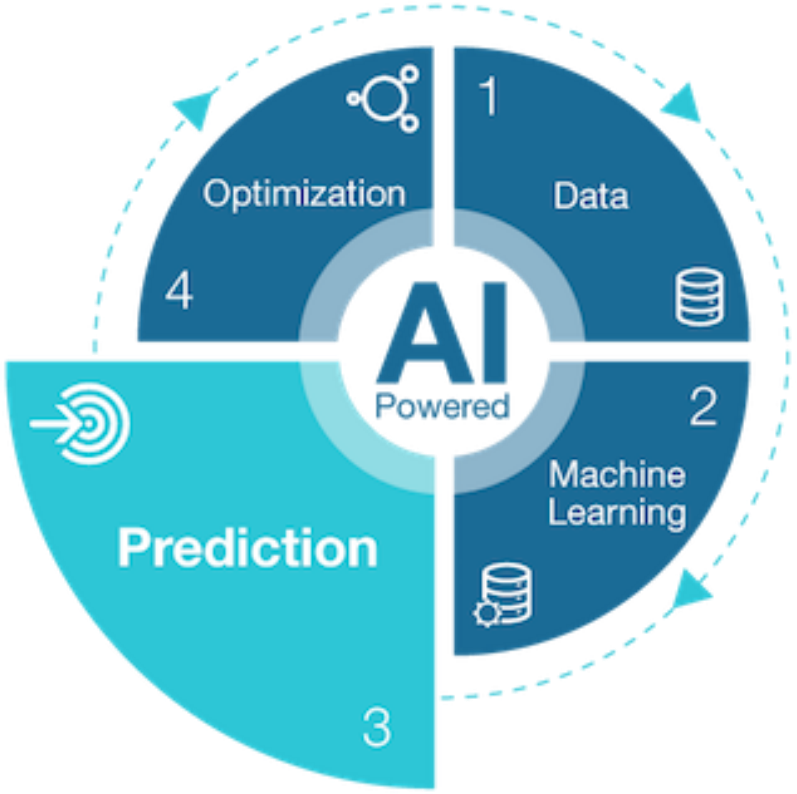
```
stregr.predict(train_x)
print("Mean Squared Error: %.4f"
      % np.mean((stregr.predict(train_x) - train_y1) ** 2))
print('Variance Score: %.4f' % stregr.score(train_x, train_y1))
```

```
Mean Squared Error: 3.1240
Variance Score: 0.7565
```

```
np.sqrt(mean_squared_error(train_y1, stregr.predict(train_x)))
```

```
1.767470448092271
```

Prediction & Future Improvement



Further Improvement:

- Impute the missing values for Item weight based on the relationship with other column values.
- Customized feature encoding for every categorical variable.
- Parameter tuning for the final ensemble model.

mengqi0128@126.com

THANK YOU

2019.04.03