Predictive Analytics Modeling Exercise - Ruonan Feng

Data Exploration

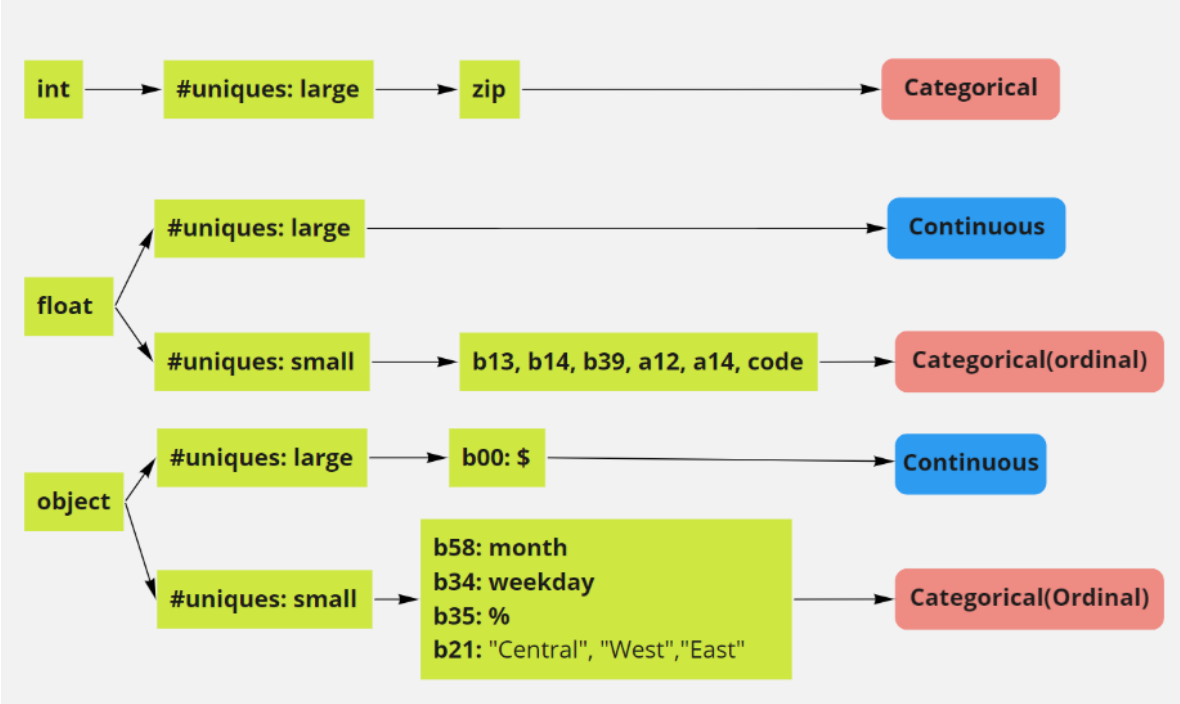
A description of your modeling process and design choices, including any data preparation, exploratory data analysis, feature manipulations, model tuning and assessment. State any assumptions you made along the way.

Sample Size	
Learning	12073
Test	5366

Features

There are 79 variables in total.

According to the number of unique values, datatype and feature content, there are 68 continuous variables and 11 categorical variables.



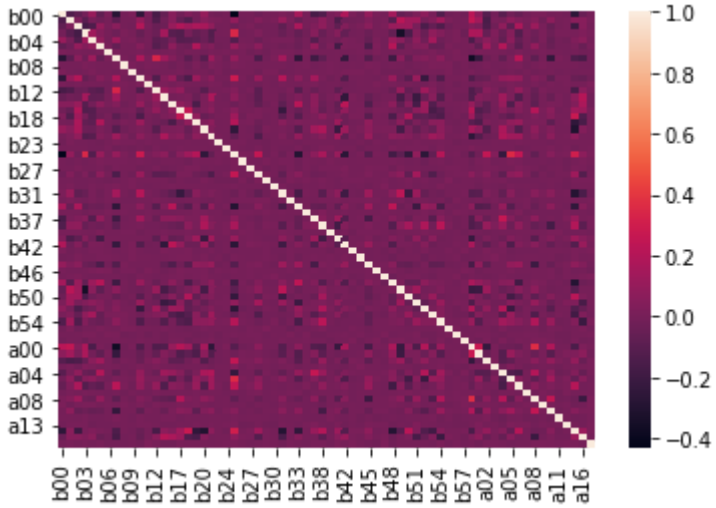
Assumption: How to classify categorical variables highly depends on the business(meaning of feature). Since we don't have enough information, so I categorize variables according to what I have.

Continuous Variables

Continuous Variables have different ranges and are roughly in normal distribution.

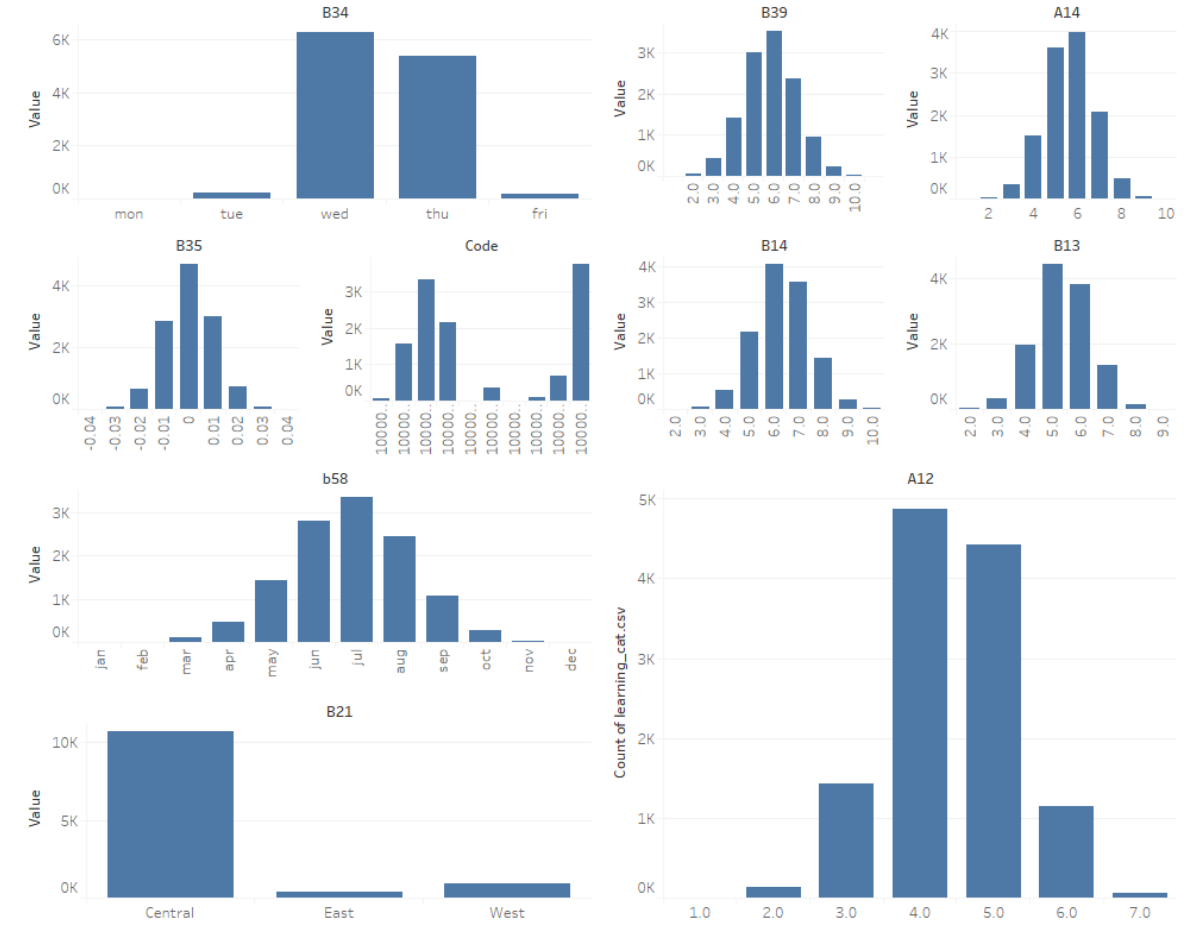
id	b52	b07	b47	a00	b32	b12	b01	b27	b15	b24	a05	b45	b23	b26
1000-46	-64.384522	851.0	2.945763	480.0	-2.312037	43.836931	14.838814	9.230083	34.430043	4.498217	15.928078	-24.237362	-5.854463	-3.476818
1001-29	37.666092	656.0	-8.700703	712.0	1.431917	-31.946287	-2.164662	10.246129	-2.435067	-0.691180	41.179466	-16.974943	-2.419538	-6.238686
1001-77	-69.160812	692.0	-2.637349	355.0	-12.691694	-46.829226	-8.230810	12.030270	4.282571	1.887015	21.485363	12.452773	-2.970784	-1.274113
1002-73	63.461019	423.0	-6.146369	641.0	-7.766379	-11.581529	31.781108	9.994060	-4.624934	3.040520	14.094273	8.236477	-3.949114	-9.482016
1002-88	7.531435	555.0	-9.576703	603.0	-5.759907	19.214385	2.874900	12.143091	-20.854148	-1.169207	4.508749	-9.490450	-3.051016	-3.836641

Draw a heatmap for feature correlation matrix, there's no obvious correlation between them. All of correlation coefficient are less than 0.8.

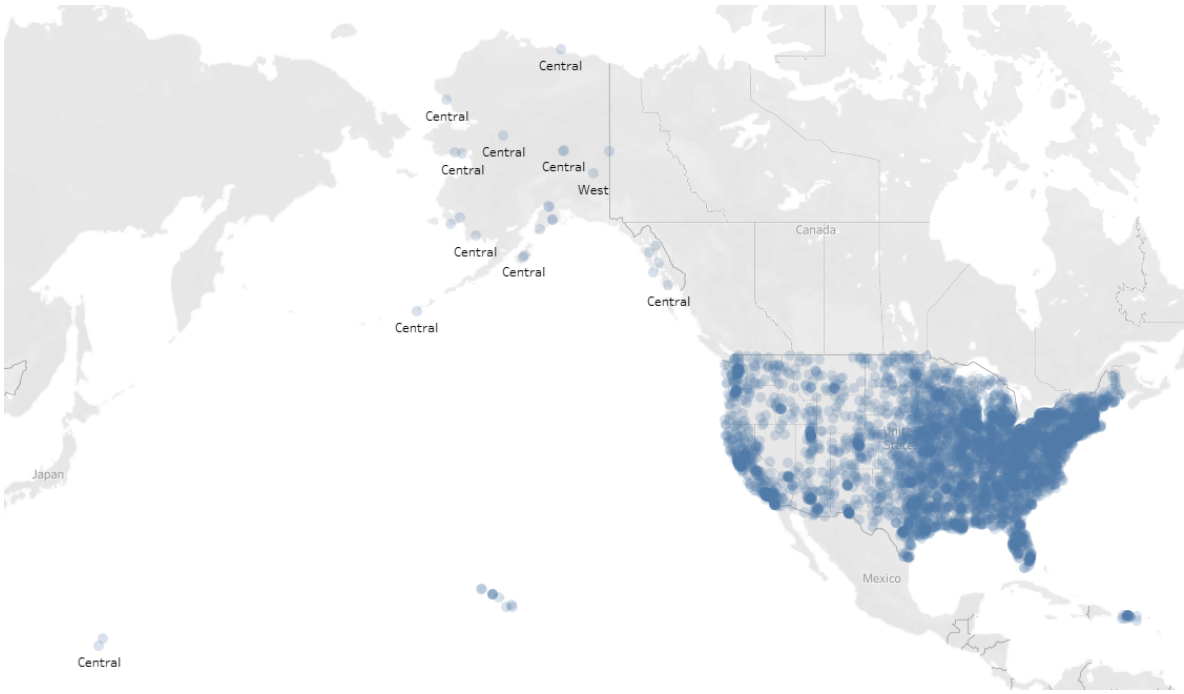


Categorical Variables

There are 11 categorical variables. The graph below shows the distribution of them except "zip".



Map the data by zip code on the map. I found feature b21(Central, East, West) doesn't related to location.



Label

The target variable is binary 0-1 which is very imbalanced. In the learning data, 10806 are 0 and 1267 are 1.

In practical, changing the data distribution will influence the model performance. And even tree models are good for imbalanced data, so I won't change data distribution.

Preprocessing

Cleaning

On both learning and test set, I did following cleaning:

	cleaning	Note
b58: month	regularize varies month format to "jan",'feb'...	depends on business, it might be allocated to 4 seasons
b34: weekday	regularize varies weekdays format to "mon", 'tue'	
b35: % percentage	remove "%" and convert to float	There are only 9 unique values so treat it as categorical value but it depends on business.
b00: $ money amount	remove "$" and space	
zip: zip code	discretize to state	Discretization: zip code has high cardinality. It can also be transformed by city size(big city, town...), location(East, West,...), etc. It depends on business and we can also compare the feature importance to target between different transformed features.
Other continuous value	/	Since I don't want to lose information or change data distribution, I didn't scale the features and will avoid scale-sensitive models like SVM.

Pictures below show the changes after cleaning:

Before:

id	b58	b34	b35	b21	b13	b14	b39	a12	a14	code	zip	b00
1000-46	jun	Wednesday	0.01 %	Central	5.0	7.0	3.0	5.0	7.0	1000002.0	78931	$ -179.05
1001-29	september	Wed.	-0.0	Central	4.0	6.0	5.0	5.0	4.0	1000009.0	2258	67.7
1001-77	Aug.	tue	0.01 %	Central	5.0	7.0	4.0	2.0	4.0	1000002.0	33871	$ 405.09
1002-73	sep.	thu	0.01%	Central	6.0	7.0	5.0	5.0	6.0	1000003.0	12588	951.36
1002-88	jul	thu	0.0	East	5.0	6.0	7.0	6.0	6.0	1000007.0	34260	106.82

After:

id	b58	b34	b35	b21	b13	b14	b39	a12	a14	code	zip	b00
1000-46	jun	wed	0.01	Central	5.0	7.0	3.0	5.0	7.0	1000002.0	TX	-179.05
1001-29	sep	wed	-0.00	Central	4.0	6.0	5.0	5.0	4.0	1000009.0	NaN	67.70
1001-77	aug	tue	0.01	Central	5.0	7.0	4.0	2.0	4.0	1000002.0	FL	405.09
1002-73	sep	thu	0.01	Central	6.0	7.0	5.0	5.0	6.0	1000003.0	NY	951.36
1002-88	jul	thu	0.00	East	5.0	6.0	7.0	6.0	6.0	1000007.0	FL	106.82

Missing Value

- **Drop**

Look at the categorical variable information:

	b58	b34	b35	b21	b13	b14	b39	a12	a14	code	zip
count	12073	12073	12073	12073	12073	12073	12073	12073	12073	12073	12073
unique	13	6	11	4	9	10	12	8	11	11	52
top	jul	wed	0.01	Central	5.0	6.0	6.0	4.0	6.0	1000009.0	nan
freq	3354	6277	3000	10723	4424	4049	3530	4862	3975	3771	7074

About 70% of zip code information is missing after being transformed to state. It might because users provided fake zip code. Since this row lost too much information, "zip" column was dropped.

- **Fill**

According to data info, few values are missing. Less than ~1.6% for each feature are missing and ~1.2% samples have missing value. Since the continuous variables are in normal distribution and categorical variables' mode are significant, I filled missing values by mean and mode.

Encoding

Assumption: Without enough background knowledge, I assume 10 categorical variables are ordinal.

So categorical variables are encoded by integers.

Standardization and Outliers

Since tree methods are not very sensitive to outliers and feature scale, in order to keep enough information as much as possible, I didn't standardized features or remove outliers.

Train Models

For model comparison, I stratified sampled 20% learning data as validation set.

	size	note
Training set	9604	copy and train for different models
Validation set	2469	evaluate models
Test	5366	for prediction

SVM (Baseline)

preprocessing

SVM might be sensitive outliers and feature scale. So before training SVM, data should be normalized and outliers should be remove

- Normalization - MinMaxScaler

  But, after doing cross validation for with/ without normalization, normalization makes it worse.

with normalization	without normalization
0.5	0.835

- Outliers: According to z-score, there are 18 sample out of 3.5σ.

  But, after cross-validation, removing outliers makes the performance worse.

So, for this data, no extra preprocessing needed for SVM.

Training

Since SVM is very slow on large data set, I trained a classifier using polynomial kernel as a baseline.

The baseline AUC on SVM classifier is ***0.835***.

Random Forest

Grid Search

Tuning hyperparameters by gridsearch with 2-fold cross validation, the final parameters are:

```
{'n_estimators': 1800,
 'criterion': 'entropy',
 'min_samples_split': 5,
 'min_samples_leaf': 16,
 'max_features': 'sqrt',
 'max_depth': 24,
 'bootstrap': False}
```

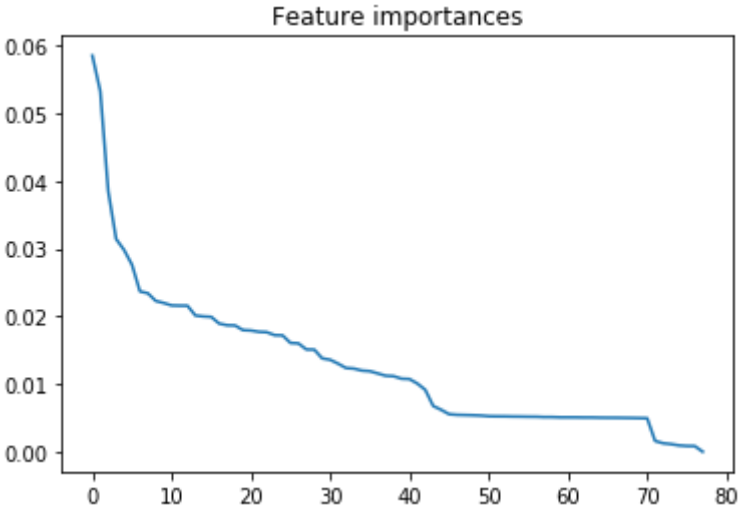
With these hyperparameters, Random Forest classifier gets ***0.938*** AUC.

Feature importance

Top 5 important features:

1. 51 (0.058427)
2. 55 (0.053178)

3. 2 (0.038479)
4. 47 (0.031412)
5. 66 (0.029803)



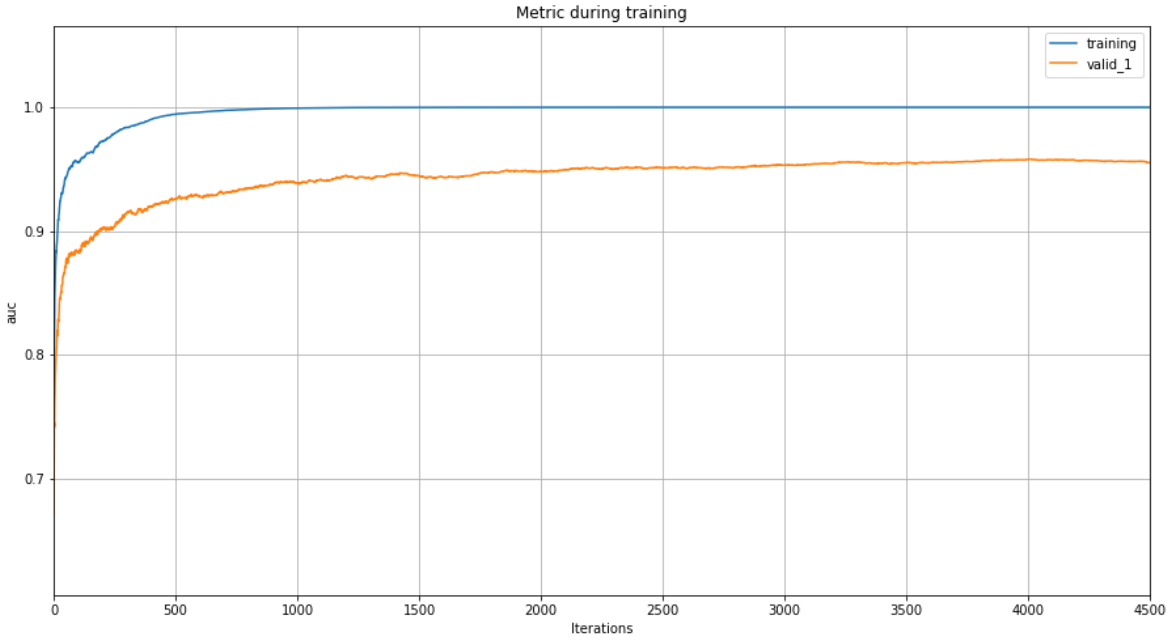
LightGBM

Grid Search

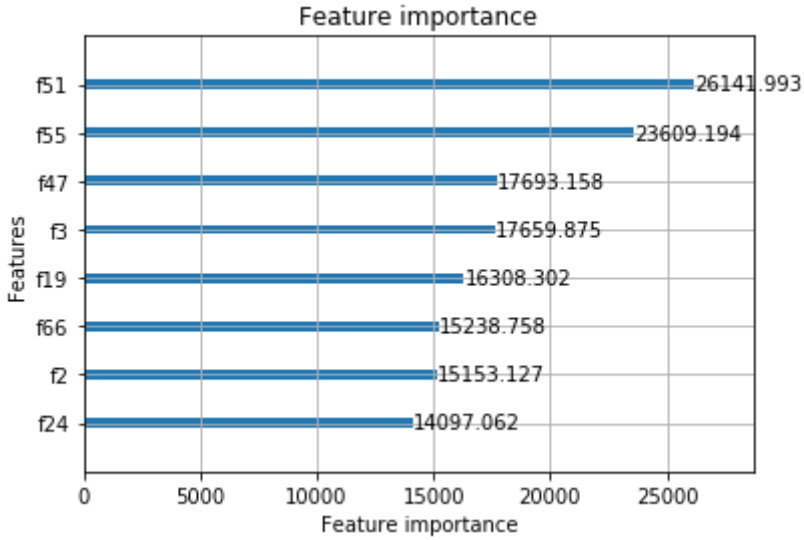
Tuning hyperparameters by grid search with 5-fold cross validation, the final parameters are:

```
{'cat_smooth': 87,
 'max_cat_threshold': 53,
 'min_data_per_group': 966,
 'learning_rate': 0.08,
 'num_iterations': 4499,
 'scale_pos_weight': 8,
 'colsample_bytree': 0.7699590061293953,
 'min_child_samples': 283,
 'min_child_weight': 10.0,
 'num_leaves': 559,
 'max_bin': 1357,
 'max_depth': 14,
 'min_data_in_leaf': 670,
 'boosting': 'dart',
 'reg_lambda': 3,
 'bagging_fraction': 0.6,
 'feature_fraction': 0.6,
 'scoring': 'roc_auc',
 'metric': 'auc',
 'objective': 'binary'}
```

With these hyperparameters, Random Forest classifier gets ***0.956*** AUC.



Feature Importance



The top features are similar to Random Forest. The most significant 2 variables are:

id	b40	b54
1000-46	-0.344525	-13.556644
1001-29	1.567580	76.176355
1001-77	0.843830	12.095990
1002-73	3.289492	-101.864166
1002-88	2.523994	5.764182

Evaluation

Metrics

Recall and Precision are not a good metrics for imbalanced data.

F1-score is a balance for both. So F1-score could be a metrics to evaluation classifiers.

AUC: An ROC curve plots TPR vs. FPR at different classification thresholds. AUC measures the entire two-dimensional area underneath the entire ROC curve. AUC provides an aggregate measure of performance across all possible classification thresholds.

Compare

Performance

	AUC on Validation Set	speed
SVM(Baseline)	0.878	very slow
Random Forest	0.951	fast
LightGBM	0.977	very fast

Since Validation set was not used for training, the AUCs above are expected performance for the prediction on test set.

Model Comparison

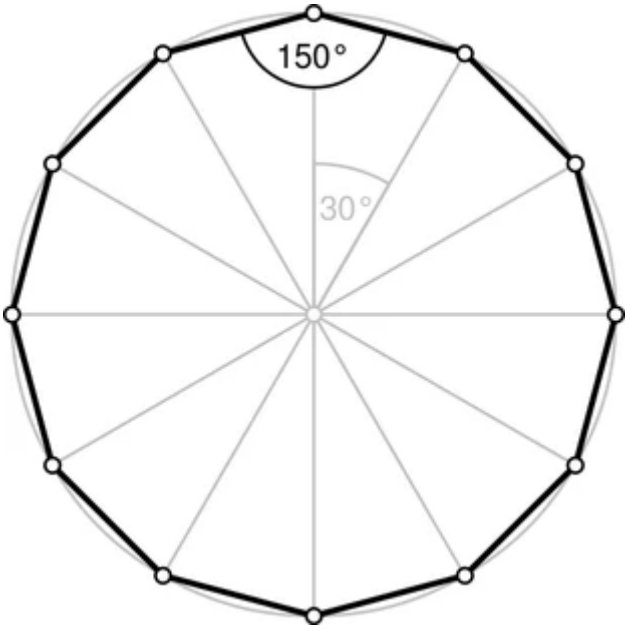
	feature	Strength	Weakness
SVM	support vectors and kernel	1. good for small data 2. Good generalization	1. slow on large data set 2. complicated for multi-class classification
Random Forest	ensemble - bagging	1. fast 2. good performance 3. parallel 4. work well on large features 5. can provide feature importance	1. overfit on data with too much noise 2. variables with too many unique values may influence the performance
LightGBM	ensemble - boosting	1. fast 2. good performance 3. parallel 4. work well on large features 5. can provide feature importance 6. save memory 7. can handle categorical variables	too much hyperparameters to tune

Other Thoughts

	20 times greater	20 time smaller
SVM	too slow to train	might be faster and have good performance on small data set
Random Forest	Better performance	Performance will be better than lightGBM
LightGBM	Better performance, faster	might be overfitting

Future Work

1. Ensemble Lgbm and Random Forest classifiers. The performance might be improved.
2. expand features
    - Variable month and weekday are periodic. Distance between December-January vs January-February should be same. If encoding by 1 to 12, December-January distance is 11 and January-February distance is 1. A better to encode: mapping 12 moths on a circle wit same distance and using coordinates to represent months.

      This way keeps periodic information but makes model less interpretable. However, it worth trying.
    - Zip: Translate zip code to states, location, city size, etc. Several features are generated and might be helpful to provide more information. Missing Zip code will be filled by NAN.
3. Other way to handle missing values

   To know how much does missing value influence the performance, do cross-validation on the data with different filling strategies: fill by mean/mode, fill by NAN, drop missing records.
4. compare outliers

   To know how much does outlier influence the performance, do cross-validation on the data with/ without outliers.
5. imbalanced data

There are several techniques to solve imbalance data, like down-sampling, up-sampling, etc. But down-sampling will lose information and up-sampling can easily be overfitting. Actually tree-based methods are good enough to handle imbalanced data. And if we do sampling, the distribution of data will be changed which maybe not good for training.

But I have read a method before and is worth trying in the future:

Do sampling on larger class and concatenate with smaller class to prepare multiple datasets. Train weak classifiers on these dataset separately and ensemble these classifiers. This way will lose model interpretability.