CS-418 Project Report

In-vehicle Coupon Recommendation System

Group-22
Jayanth Appalla (UIN: 668824973)
Nancy Pitta (UIN: 672134497)
Sridhar Addagatla (UIN: 653709811)
Sai Nirmal Morampudi (UIN: 650007349)

Introduction

When applying machine learning models to domains such as customer behavior analysis and demand forecasting, along with predicting the decision, one would also like to understand the various factors that led to this decision and what this decision says about the data itself. Coupons help businesses grow market share, increase sales volume, sell faster, cultivate loyal customers, and drown out competitor advertising. Hence, understanding the target customer base to push the coupons would enable a business to grow faster and increase revenue.

The goal of this project is to predict whether a driver would accept the coupon or not while driving given various external conditions at the time such as weather, time of the day, number of passengers, etc. This is a classification problem where the outcome would be acceptance or rejection of the coupon. The data was collected by conducting a survey on amazon mechanical Turk.

Some of the questions this project uncovers are the trends and patterns of attributes with respect to the target variable using various EDA methods. In addition to this, we have explored how different machine learning models such as logistic regression, SVM, decision trees perform on this dataset.

Data

Data collection

The dataset for this problem is taken from the UCI repository. The collected dataset was raw and unprocessed, so various pre-processing techniques and transformation techniques are used to make the dataset suitable for modeling.

Data description

This dataset contains a single CSV file with a total of 12684 instances. The given dataset was a balanced dataset with 56% positive outcomes(coupon acceptance) and 43% negative outcomes(coupon rejection). One peculiarity of this dataset is that it has all categorical features. There are a total of 26 features in this dataset with the target feature being “Coupon_accepted” with two values for it (0-rejection, 1-acceptance). The dataset contains contextual attributes like weather, destination, temperature, time of the day along with user-specific information like gender, education, income, occupation, passenger, marital status, etc. In addition to this, coupon-specific information.

The attributes are summarized in the following table:

USER ATTRIBUTES	**CONTEXTUAL ATTRIBUTES**	**COUPON ATTRIBUTES**
Gender	destination	coupon
Age	passenger	expiration
Marital_status	Weather	
Has_children	temperature	
Education	time	
Occupation	toCoupon_GEQ5min	
income	toCoupon_GEQ15min	
bar	toCoupon_GEQ25min	
coffeehouse	direction_same	
carryAway	direction_opp	
restaurantlessthan20		

restaurant20To50		

The following shows a few instances of the dataset:

```
data.head()
```

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	has_children	education	occupation	income	car	
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	1	Some college - no degree	Unemployed	$37500 - $49999	NaN	ne
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	1	Some college - no degree	Unemployed	$37500 - $49999	NaN	ne
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	1	Some college - no degree	Unemployed	$37500 - $49999	NaN	ne
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	1	Some college - no degree	Unemployed	$37500 - $49999	NaN	ne
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	1	Some college - no degree	Unemployed	$37500 - $49999	NaN	ne

Data Preprocessing

Dropping the irrelevant columns

We found a few attributes that contained redundant information. So we removed those features. In particular, the attribute “car” contained 99% null values and the attribute “toCoupon_GEQ5min” contained only one unique value which couldn’t be used for prediction. So, we dropped those attributes. In addition to this, the attributes “direction_opp” and “direction_same” were highly correlated. Hence, we used only one attribute and dropped the other.

Addressing the missing values

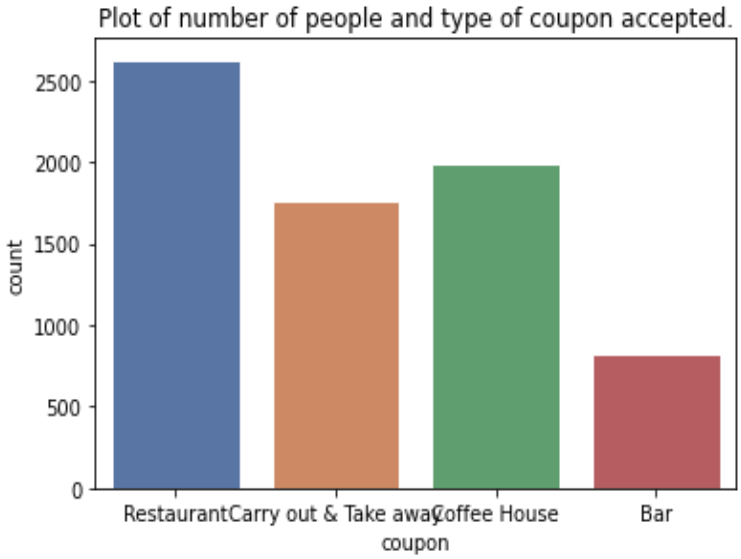
There are 5 attributes (“CoffeeHouse”, “Bar”, “CarryAway”, “RestaurantLessThan20”, “Restaurant20To50”) with missing values in this dataset. Each of these attributes contains 1-2% missing values and we used Knn_imputer algorithm to impute the missing values. As we found this method was giving better results than mode.

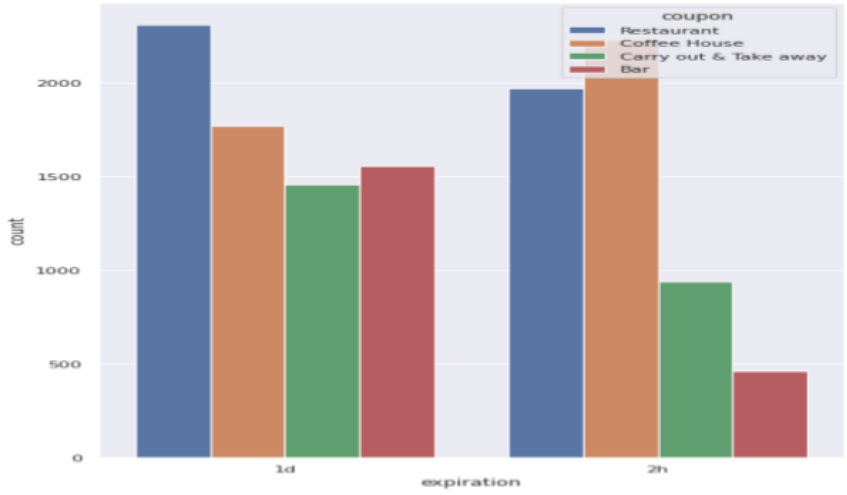
Insights from the Data

After data cleaning, we explored the data using different types of libraries in python. Different charts such as bar charts, line plots, and correlation plots were used to analyze the data and find some insights into the data. Libraries such as pandas, seaborn, and matplotlib were used to build the plots.

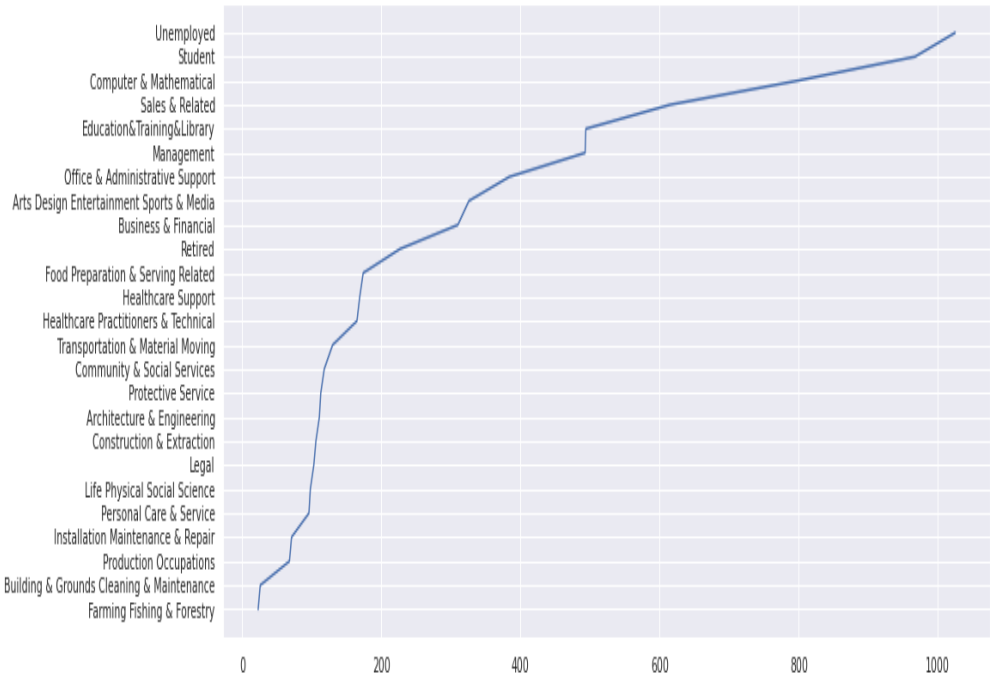
Some of the interesting insights we found in the dataset are as follows.

1. Restaurant coupons have the highest acceptance rate because most of the restaurant coupons are valid for a longer duration. This can be observed in the following plots.

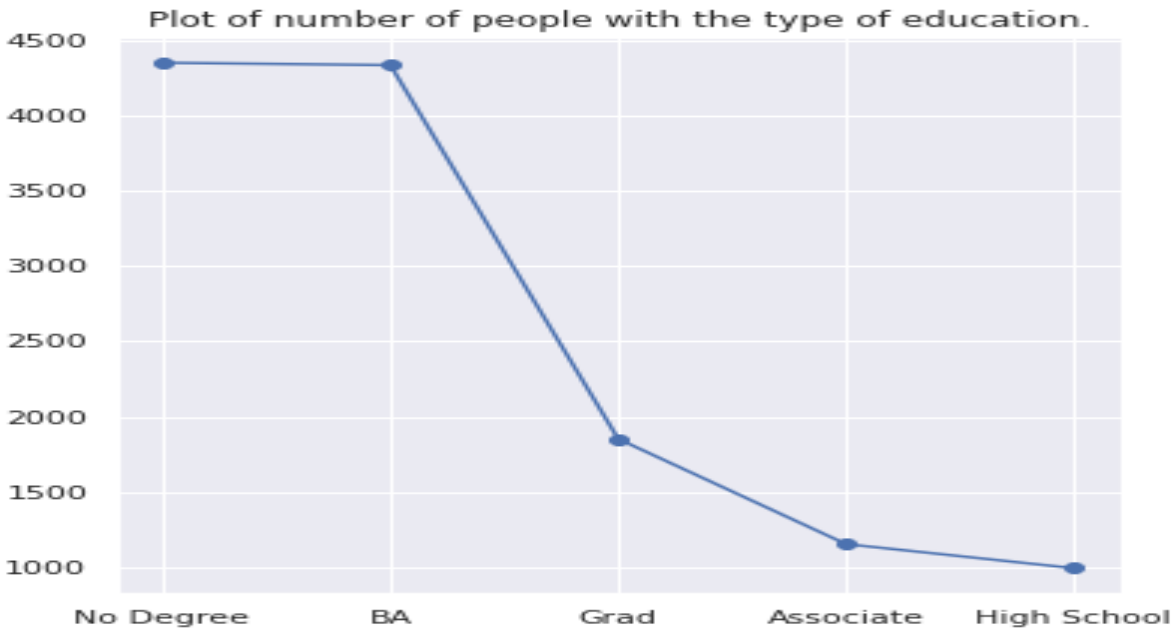




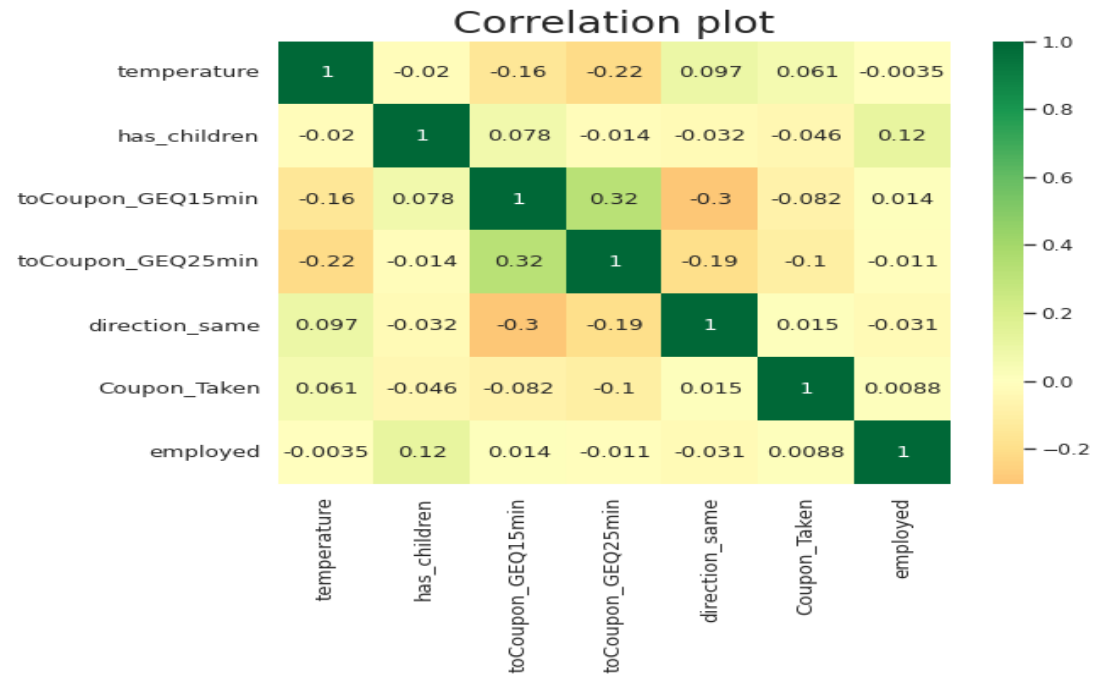
2. More coupons were accepted by students and Unemployed people than people with occupations such as farming, cleaning. This behavior was expected and the dataset corroborates with the assumptions.



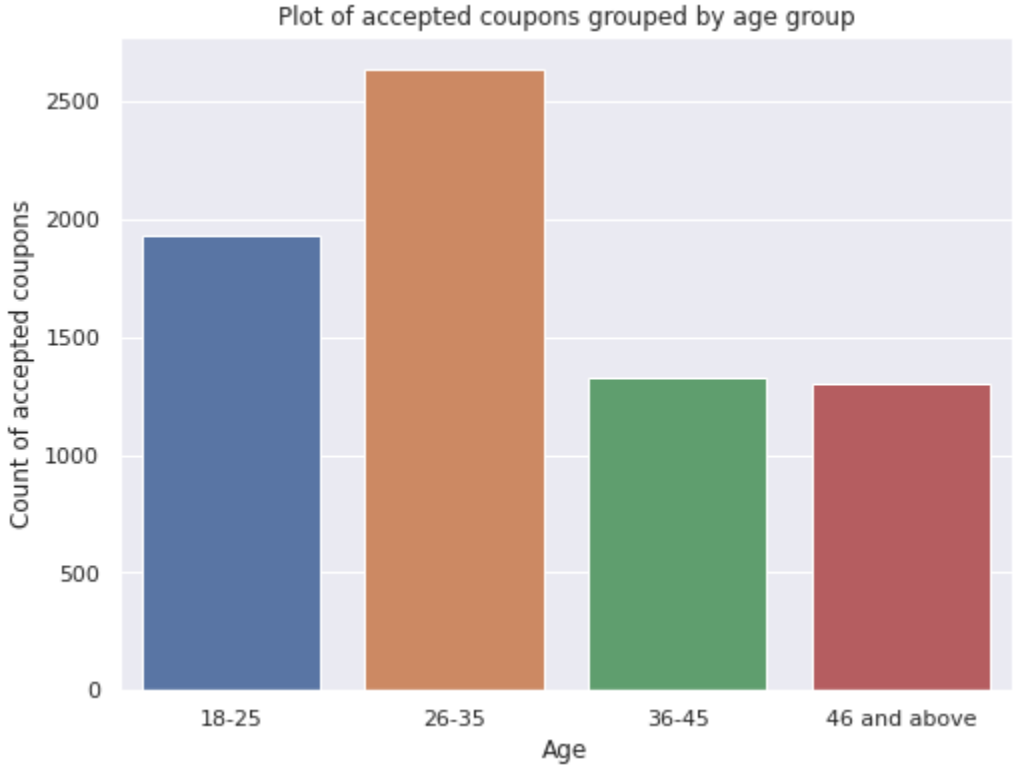
3. People with no degree accepted more coupons than people with higher education.



4. From the following correlation plot, we can observe that features are not highly correlated with each other and with the target variable. This means there are no influential attributes in the dataset.



5. The younger people and middle-aged people accepted more coupons than older age people.



These insights are used in the following "Data Transformation" step to encode the categorical features into numerical form.

Data Transformation

All the features of this dataset are categorical and these can be classified into ordinal, nominal, and binary. We have used separate data encoding techniques for each of these categories.

Ordinal Attributes:

The following ordinal attributes are encoded to numerical form using the analysis we observed in the above EDA step.
For example,
For the “Age” attribute, higher numerical values are given to the age group which accepted more coupons.

ORDINAL ATTRIBUTES	VALUES
CarryAway , CoffeeHouse , Bar, Restaurant20To50 , RestaurantLessThan20 , income , education , time	never, less1 , 1 to 3 , 4 to 8 , greater than 8
age	below21 , 21 , 26 , 31 , 36 , 41 , 46 , 50plus
income	Less than $12500 , $12500-$24999 , $25000-$37499,$37500-$49999, $50000-$62499 , $62500-$74999 , $75000-$87499 , $87500 - $99999 , $100000 or More
education	some college - no degree , Bachelor's degree , Associates degree , high school graduate , graduate degree(Masters or Doctorate) , Some High School
Time	7AM , 10AM , 2PM , 6PM , 10PM

Nominal Attributes:

The following nominal attributes are transformed using the one-hot encoding method. As there is no order among the values of these attributes, we have used the one-hot encoding method.

NOMINAL ATTRIBUTES	VALUES
destination	No Urgent Place , Home, Work
passanger	Alone, Friend, Family
weather	Sunny , Rainy , Snowy
coupon	Restaurant , Bar , CarryAway & Takeout,CoffeeHouse
MaritalStatus	Single, Unmarried Partner, Married Partner
gender	Male, Female
expiration	1d, 2h

Frequency Encoding:

The dataset contains one special attribute “occupation” which contained 28 unique values. There was no order in these values, So we can not perform label encoding. The dimensionality would drastically increase if we do the one-hot encoding of this feature. So, we have performed the frequency encoding method to transform this attribute.

ATTRIBUTE	FEATURES
Occupation	Unemployed , Architecture & Engineering , Student , Education&Training&Library, Healthcare Support , Healthcare Practitioners & Technica , Sales & Related, Management , Arts Design Entertainment, Sports & Media , Computer & Mathematical , …..

Methods

Data Reduction

After data transformation, the dataset contains 37 features and the dataset has become very sparse. It is very difficult to properly fit a model given the number of training examples and number of features. So, with the following two data reduction techniques, we tried to make the sparse dataset dense.

Principal component analysis (PCA)

PCA is a technique used for reducing the dimensionality of datasets while minimizing information loss. The principal components obtained are linearly uncorrelated and capture the maximum variance in the data.

```
pca.explained_inertia_*100
```

```
array([1.16065442e+01, 8.69028452e+00, 7.90483804e+00, 6.86564515e+00,
       5.06348464e+00, 4.74856544e+00, 4.40064426e+00, 4.33357657e+00,
       4.21557058e+00, 4.01051902e+00, 3.74773139e+00, 3.58490172e+00,
       3.43393502e+00, 3.22473450e+00, 2.90582789e+00, 2.88636315e+00,
       2.70950137e+00, 2.56289329e+00, 2.43711827e+00, 2.03086023e+00,
       1.81623289e+00, 1.68371290e+00, 1.44962406e+00, 1.30297431e+00,
       1.21794864e+00, 6.83464962e-01, 4.82503022e-01, 2.28898723e-30,
       7.02306421e-32, 7.02306421e-32, 7.02306421e-32, 7.02306421e-32,
       7.02306421e-32, 7.02306421e-32, 7.02306421e-32, 7.02306421e-32,
       7.02306421e-32])
```

```
sum(pca.explained_inertia_[:20])
```

```
0.9136353921308825
```

```
sum(pca.explained_inertia_[:25])
```

```
0.9883403201594441
```

Analysis:

- The top 25 principal components could explain 98.8% variability of the original data. So, we have taken 25 principal components as the feature space for further analysis.
- Each of the principal components holds less information. This explains the sparsity in the dataset.

Recursive Feature Elimination (RFE)

RFE is a wrapper-type feature selection algorithm. This means that a different machine learning algorithm is given and used in the core of the method, is wrapped by RFE, and used to help select features. We have used the “Random forest” classifier in RFE to select the top 20 features from the dataset.

Following shows selected features:

```
print (data.columns[rfe.support_])
print (len(data.columns[rfe.support_]))
```

```
Index(['temperature', 'time', 'expiration', 'gender', 'age', 'has_children',
       'education', 'occupation', 'income', 'toCoupon_GEQ15min',
       'destination_No Urgent Place', 'passanger_Friend(s)', 'weather_Sunny',
       'coupon_Bar', 'coupon_Carry out & Take away', 'coupon_Coffee House',
       'coupon_Restaurant(<20)', 'maritalStatus_Married partner',
       'maritalStatus_Single', 'maritalStatus_Unmarried partner'],
      dtype='object')
20
```

Analysis from Data reduction:

After the data reduction step, the following feature spaces are used in the data modeling phase.

- All the features were obtained from the feature transformation step i.e. 37 features.
- Features obtained from PCA i.e. 25 features.
- Features obtained from RFE technique i.e 20 features.

These 3 feature spaces are used in the modeling phase to understand how different models are going to perform on the dense data compared to sparse data.

Analysis

Machine Learning Algorithms used:

Logistic Regression

Logistic regression is a classification algorithm used to predict a binary outcome based on the relationship between independent variables. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables.

Implementation:

- We have split the training and test data into an 80-20 ratio.
- The training data was used in the K-Fold cross-validation to find the best hyperparameters for the data.
- The hyperparameter search space used is {'C': [0.02,0.08,0.1,2]}
- This algorithm was implemented on the 3 feature spaces obtained in the previous step.

Following shows the results:

logistic regression with PCA features

	params	mean_test_score	rank_test_score
0	{'C': 0.02}	0.656647	1
1	{'C': 0.08}	0.656154	3
2	{'C': 0.1}	0.656154	2
3	{'C': 2}	0.656055	4

logistic regression with 37 features

	params	mean_test_score	rank_test_score
0	{'C': 0.02}	0.665517	3
1	{'C': 0.08}	0.666108	1
2	{'C': 0.1}	0.665122	4
3	{'C': 2}	0.665812	2

logistic regression with RFE

	params	mean_test_score	rank_test_score
0	{'C': 0.02}	0.658617	2
1	{'C': 0.08}	0.658025	3
2	{'C': 0.1}	0.658715	1
3	{'C': 2}	0.657730	4

Analysis:
The model performed similarly on all 3 feature spaces. The highest mean cross-validation accuracy was achieved when the model was trained with all 37 features.

Support vector machine

A support vector machine (SVM) is a supervised machine learning model that can be used for both regression and classification problems. It is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well

The implementation steps done in this model are similar to the steps done in the previous algorithm with the following hyper-parameter search.

- {'C': [0.1, 1, 4], 'gamma': [0.1, 1]}
- "**rbf**" kernel is used to increase the model complexity and to fit the data well.

Following are the results that obtained:

SVM with all features 37

	params	mean_test_score	rank_test_score
0	{'C': 0.1, 'gamma': 0.1}	0.654085	3
1	{'C': 0.1, 'gamma': 1}	0.565586	6
2	{'C': 1, 'gamma': 0.1}	0.689367	1
3	{'C': 1, 'gamma': 1}	0.624520	5
4	{'C': 4, 'gamma': 0.1}	0.683749	2
5	{'C': 4, 'gamma': 1}	0.631321	4

SVM with pca features

	params	mean_test_score	rank_test_score
0	{'C': 0.1, 'gamma': 0.1}	0.648665	3
1	{'C': 0.1, 'gamma': 1}	0.566867	6
2	{'C': 1, 'gamma': 0.1}	0.671825	1
3	{'C': 1, 'gamma': 1}	0.596827	4
4	{'C': 4, 'gamma': 0.1}	0.659999	2
5	{'C': 4, 'gamma': 1}	0.591602	5

SVM with RFE

	params	mean_test_score	rank_test_score
0	{'C': 0.1, 'gamma': 0.1}	0.663052	3
1	{'C': 0.1, 'gamma': 1}	0.569331	6
2	{'C': 1, 'gamma': 0.1}	0.695181	2
3	{'C': 1, 'gamma': 1}	0.648466	5
4	{'C': 4, 'gamma': 0.1}	0.696659	1
5	{'C': 4, 'gamma': 1}	0.656054	4

Analysis:

- SVM with a non-linear kernel performed better than logistic regression.
- Even when using SVM, models learned on all 3 feature spaces showed similar results.

Decision Tree

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression.

To find the best hyper parameters for this algorithm we have used used k-fold cross-validation in the following search space:
{'criterion': ['gini', 'entropy'], 'max_depth': [5, 10, 20]}

Following results are obtained:

Decision tree with full features (37)

	params	mean_test_score	rank_test_score
0	{'criterion': 'gini', 'max_depth': 5}	0.656845	2
1	{'criterion': 'gini', 'max_depth': 10}	0.656547	3
2	{'criterion': 'gini', 'max_depth': 20}	0.627967	5
3	{'criterion': 'entropy', 'max_depth': 5}	0.657634	1
4	{'criterion': 'entropy', 'max_depth': 10}	0.651324	4
5	{'criterion': 'entropy', 'max_depth': 20}	0.622744	6

Decision tree with PCA features

	params	mean_test_score	rank_test_score
0	{'criterion': 'gini', 'max_depth': 5}	0.639301	1
1	{'criterion': 'gini', 'max_depth': 10}	0.631318	3
2	{'criterion': 'gini', 'max_depth': 20}	0.606484	5
3	{'criterion': 'entropy', 'max_depth': 5}	0.638611	2
4	{'criterion': 'entropy', 'max_depth': 10}	0.630530	4
5	{'criterion': 'entropy', 'max_depth': 20}	0.604218	6

Decision tree with RFE features

	params	mean_test_score	rank_test_score
0	{'criterion': 'gini', 'max_depth': 5}	0.655464	2
1	{'criterion': 'gini', 'max_depth': 10}	0.643245	3
2	{'criterion': 'gini', 'max_depth': 20}	0.632008	5
3	{'criterion': 'entropy', 'max_depth': 5}	0.656154	1
4	{'criterion': 'entropy', 'max_depth': 10}	0.641965	4
5	{'criterion': 'entropy', 'max_depth': 20}	0.627377	6

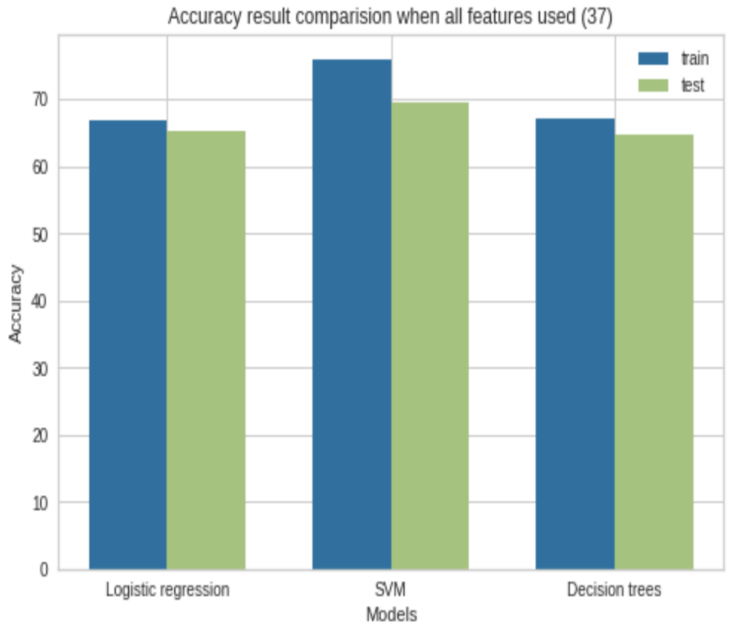
Analysis:

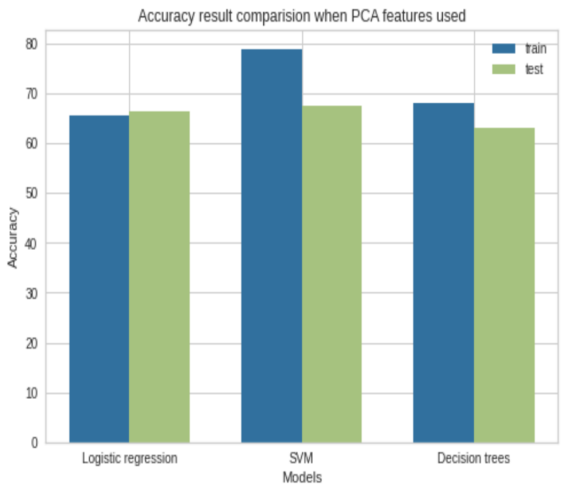
- The decision tree learned on all the features performed better than other feature spaces.
- Accuracy is comparable for all the feature spaces when used decision tree algorithm.

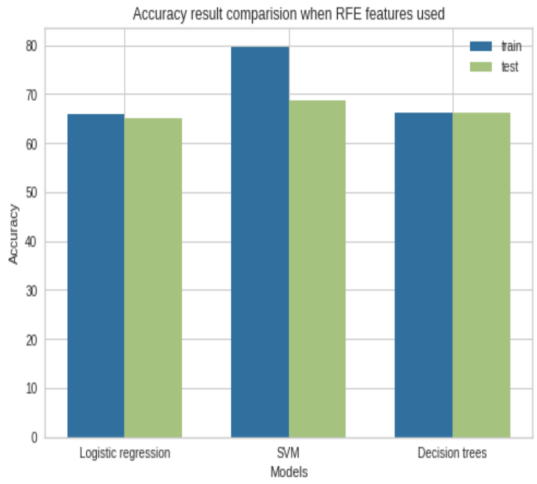
Result

Using the best performing hyper-parameters learned from the previous step, we analyzed the performance of models on test data. Since the dataset is a balanced dataset and the observed precision, recall scores were similar, we used only accuracy as the performance metric.

The following plot summarizes the training error on 80% of data and test error on 20% of data.







FEATURE SPACE	HIGHEST TEST ACCURACY
All the features obtained from Data Transformation (37)	69.49% (SVM)
Features obtained from PCA (25)	67.40% (SVM)
Features obtained from RFE (20)	68.82% (SVM)

Result Analysis and conclusion

- SVM performed much better than the logistic regression, decision tree models on all the 3 feature spaces.
- The cross-validation techniques we used to find the best hyper-parameters resulted in increasing the accuracy of test data.
- Models trained with features obtained from PCA, RFE gave similar results as compared to models trained with all the features. Thus data reduction helped us in reducing the training time while maintaining accuracy.
- From the results, we can observe that the models are underfitting because the training error and the test error are comparable for each of the models. To avoid this underfitting problem, we have taken the following steps:
    - We have increased the model complexity from linear (logistic regression) to non-linear (SVM, Decision trees)
    - We made the feature space dense by using feature reduction techniques.

The smaller size of the dataset is the biggest reason for the models to underfit. This is evident from the fact that all models got similar accuracy scores for training and testing. Thus we can conclude that the dataset could not properly represent the underlying data distribution to capture the relationship between input and output variables accurately.