The University of Chicago Booth School of Business

Machine Learning

Winter 2020

HW #2

Jiyoon Chung

Arman Bhuiyan

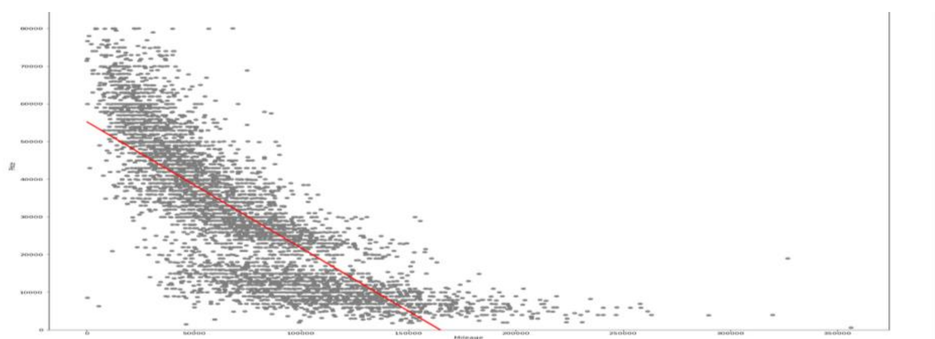
Hikaru Sugimori

Deepak Putchakayala

1.1
The data-set includes historical car transaction data with a number of different data: price, year, color, region and more

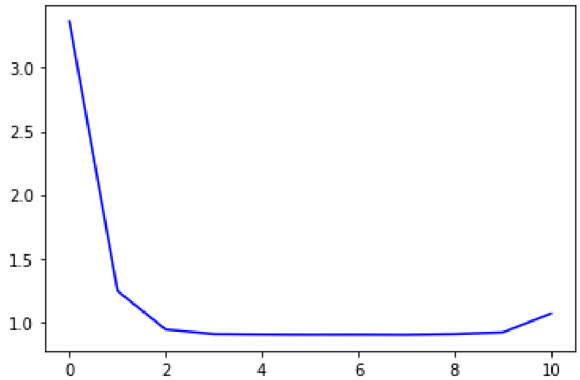
1.2
```
carsdf = pd.read_csv("UsedCars.csv")
carsnp = carsdf.values
training_set, testing_set = train_test_split(carsnp, test_size=0.25, random_state=0)
```

1.3 Linear Regression
```
lm = LinearRegression()
lm.fit(X_train, y_train)
print(lm.intercept_)
plt.figure(figsize=(20,20))
plt.scatter(X_test, y_test,  color='gray')
plt.xlabel("Mileage")
plt.ylabel("Price")
plt.plot(X_test, y_pred, color='red', label='linear Regression')
plt.gca().set_ylim(bottom=0)
plt.legend()
plt.show()
```



```
MSE = 124281428.121389
Intercept = 55438.601012
Coefficient = -0.337798
```

1.4 Polynomial Regression

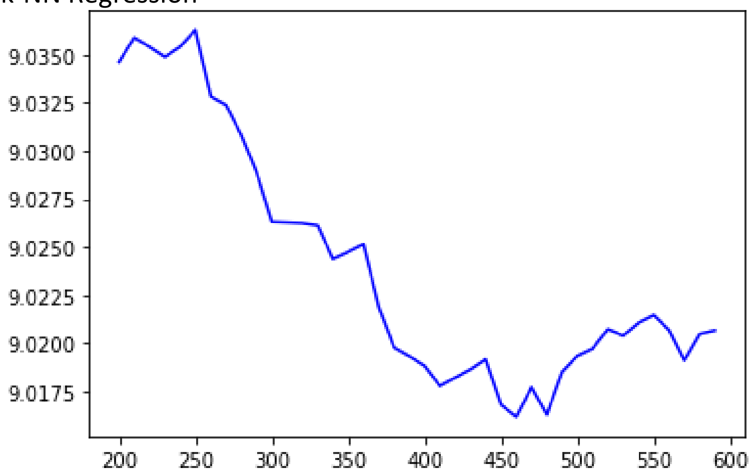


Optimal polynomial degree = 7
MSE = 89876901.721073

```
mse_plot = [0] * 11
degrees = list(range(11))


KF = KFold(n_splits=5, shuffle=True, random_state=0)
folds = KF.split(mileage_train, price_train)

for fold in folds:
   train_index = list(fold[0])
   cv_index = list(fold[1])
   x_train = mileage_train[train_index, :]
   y_train = price_train[train_index, :]
   x_cv = mileage_train[cv_index, :]
   y_cv = price_train[cv_index, :]

   for d in degrees:
      mse_temp = polynomial(d, x_cv, y_cv, x_train, y_train)
      mse_plot[d] = mse_plot[d] + mse_temp

best_degree = min(mse_plot)
for i in range(11):
   if best_degree == mse_plot[i]:
      best_degree = i
   mse_plot[i] = mse_plot[i] / 5

plt.plot(degrees, mse_plot, color='blue')
plt.show()
```

1.5

k-NN Regression



Optimal K = 460

MSE = 90161604.079432

```
mse_plot = [0] * 40
k = list(range(200, 600, 10))

KF = KFold(n_splits=5, shuffle=True, random_state=0)
folds = KF.split(mileage_train, price_train)

for fold in folds:
   train_index = list(fold[0])
   cv_index = list(fold[1])
   x_train = mileage_train[train_index, :]
   y_train = price_train[train_index, :]
   x_cv = mileage_train[cv_index, :]
   y_cv = price_train[cv_index, :]

   for i in k:
      mse_temp = kNN(i, x_cv, y_cv, x_train, y_train)
      mse_plot[(i-200)//10] = mse_plot[(i-200)//10] + mse_temp

best_k = min(mse_plot)
for i in range(40):
   if best_k == mse_plot[i]:
      best_k = i
   mse_plot[i] = mse_plot[i] / 5

plt.plot(k, mse_plot, color='blue')
plt.show()
```

k-NN function

```
def kNN(k, x_cv, y_cv, x_train, y_train):
    knn = KNeighborsRegressor(k, weights='uniform')
    y_predict = knn.fit(x_train, y_train).predict(x_cv)
    mse = mean_squared_error(y_cv, y_predict)

    return mse
```

Question 2

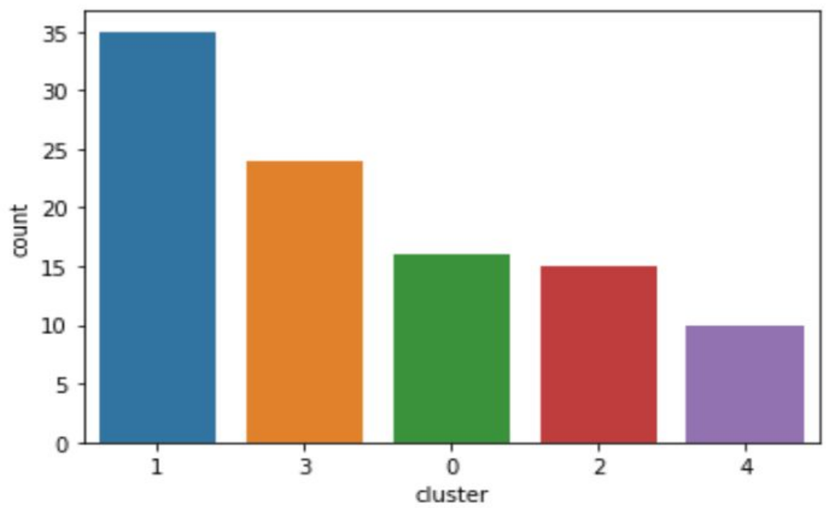
Principal component analysis (PCA) is a technique used to emphasize variation and bring out strong patterns in a dataset. PCA is essentially just a coordinate transformation. The original data are plotted on an *X*-axis and a *Y*-axis. For two-dimensional data, PCA seeks to rotate these two axes so that the new axis *X'* lies along the direction of maximum variation in the data. For this problem, we will generate and analyze cluster in the dataset using this technique.

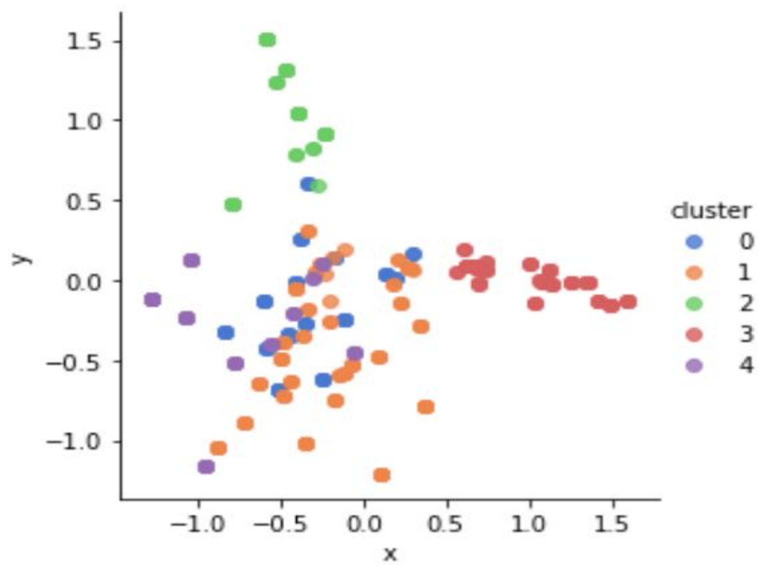
We merged the wine deals and wine transactions datasets since they shared a column, created a pivot table to show customer response successes per offer, sliced the matrix to isolate only the binary indicator column, counted the number of people who ended up in each cluster and plotted them. We also transformed the two dimensional dataset via PCA, created a new df with name, cluster membership, and coordinates, merged that dataframe, and created a scatterplot. For more details on how this was done, please see the code at the end.

```
[In [9]: averages = review_cluster(merged_df, 4)

[In [10]: averages
 Out[10]:
       Discount (%)  Minimum Qty  cluster
 True     60.639175    75.030928      1.0
 True     64.652174    79.617391      2.0
 True     47.387097    51.483871      3.0
 True     55.830508     6.000000      4.0
```

The screenshot above describes the clusters we got in terms of discount % and minimum quantity. Cluster 1 and Cluster 2 are very similar in terms of their quantity and discount, so it's likely the varietal or origins that are different. Cluster 4 takes the least discount and the least quantity, so if there are also fewer of them, these are people that don't buy much. Cluster 3 is somewhere in the middle.

Visualisations of our clusters are below. Looks like most people below to cluster 1 and 3.





8]:

is_4	min_quantity	discount
False	51.512727	59.287273
True	97.102041	60.571429

```
(Varietal
 Champagne             57
 Cabernet Sauvignon    32
 Malbec                15
 Chardonnay            14
 Merlot                13
 Prosecco              11
 Pinot Noir             7
 Espumante              6
 Pinot Grigio           2
 Name: Varietal, dtype: int64, Origin
 France          78
 Chile           24
 Italy           12
 Oregon          10
 California       9
 New Zealand      7
 Australia        6
 South Africa     6
 Germany          5
 Name: Origin, dtype: int64)
```

First cluster liked a variety of origins and varietals, driven by France and champagne

```
(Varietal
 Espumante       25
 Malbec          12
 Pinot Grigio     9
 Prosecco         6
 Merlot           4
 Champagne        1
 Name: Varietal, dtype: int64, Origin
 France          21
 South Africa    16
 Oregon           9
 Chile            5
 Australia        4
 California       1
 New Zealand      1
 Name: Origin, dtype: int64)
```

The second cluster mostly liked espumante and malbec from France and South Africa.

```
(Varietal
 Champagne    20
 Prosecco      9
 Espumante     5
 Name: Varietal, dtype: int64, Origin
 France          11
 Chile            6
 New Zealand      4
 Australia        3
 Germany          3
 Oregon           3
 California       2
 South Africa     2
 Name: Origin, dtype: int64)
```

The third cluster liked even fewer things, but mostly champagne (see above).

```
(Varietal
 Pinot Noir    37
 Prosecco       2
 Champagne      1
 Chardonnay     1
 Malbec         1
 Merlot         1
 Name: Varietal, dtype: int64, Origin
 Australia       13
 Italy           12
 France           7
 Germany          7
 California       2
 New Zealand      1
 South Africa     1
 Name: Origin, dtype: int64)
```

The fourth cluster basically only drinks pinot from Australia or Italy

The code for #2 is below.

```
def collect_data():
	wine_deals = pd.read_csv('Wine_deals.csv')
	wine_transactions = pd.read_csv('Wine_transactions.csv')
	wine_transactions['n'] = 1
	df = pd.merge(wine_deals, wine_transactions)
	df = df[['Offer #', 'Campaign', 'Customer Last Name', 'Origin', 'Varietal', 'Discount (%)', 'Minimum
Qty', 'Past Peak', 'n']]
	return df, wine_transactions, wine_deals

def pivot(df):
	# Pivot table to show customer response successes per offer
	matrix = df.pivot_table(index=['Customer Last Name'], columns=['Offer #'], values='n')
	matrix = matrix.fillna(0).reset_index()
```

```
        x_columns = matrix.columns[1:]
        cluster = KMeans(n_clusters=5)
        matrix['cluster'] = cluster.fit_predict(matrix[matrix.columns[2:]])
        matrix.cluster.value_counts()
        pca = PCA(n_components=2)
        matrix['x'] = pca.fit_transform(matrix[x_columns])[:,0]
        matrix['y'] = pca.fit_transform(matrix[x_columns])[:,1]
        matrix = matrix.reset_index()
        return matrix

def plot_clusters(matrix):
        sns.countplot(x='cluster', data=matrix, order=matrix['cluster'].value_counts().index)
        plt.show()

def merge_cluster_dataframe(matrix, wine_transactions, wine_deals):
        customer_clusters = matrix[['Customer Last Name', 'cluster', 'x', 'y']]
        df = pd.merge(wine_transactions, customer_clusters)
        df = pd.merge(wine_deals, df)
        return df

def plot_merged_dataframe(df):
        sns.lmplot('x', 'y', data=df, hue='cluster', fit_reg=False, palette='deep', legend=True, size=4)
        plt.show()

def review_cluster(df, cluster_number):
        averages = pd.DataFrame()
        for x in range(0,cluster_number):
        df[x] = df.cluster==x
        subset = df.groupby(x)[['Discount (%)', 'Minimum Qty']].mean()
        subset['cluster'] = x+1
        averages = averages.append(subset.loc[True])
        return averages

df, wine_transactions, wine_deals = collect_data()
matrix = pivot(df)
merged_df = merge_cluster_dataframe(matrix, wine_transactions, wine_deals)
plot_merged_dataframe(merged_df)
averages = review_cluster(merged_df, 4)
```