12 APRIL 2021

MACHINE LEARNING PROJECT ON

LAS VEGAS STRIP DATA SET

Name -Aman Patel

Section -KM002

REG-No - 11807788

Email - imamanpatel99@gmail.com

ABSTRACT

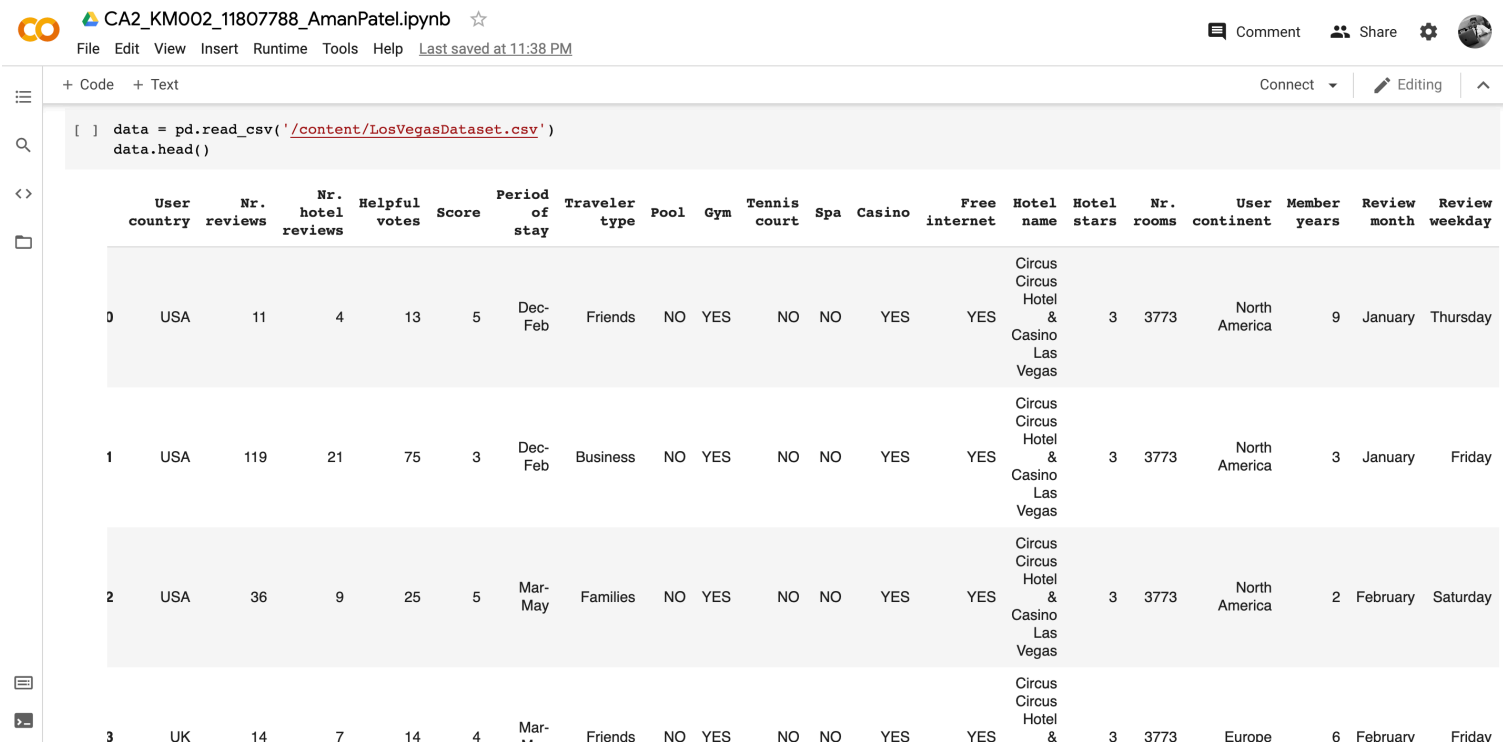
I did a project to Review Las Vegas Strip Data set machine learning in Python, almost an year back. I used Python's Scikit Learn library, along with Pandas, and Numpy on an open data set, to read and classify handwritten digits.

I used K Nearest neighbors and Random Forest tree and Support vector Classifier to achieve an accuracy of 46.5%. The process that I followed is mentioned below, along with the .ipynb file and pdf with the code. The logic behind creating a machine learning classifier for any data is very simple.

Scikit-learn only understands numbers and floats, and the number of columns for all the rows in the training and testing data sets should always be equal. Therefore, in order to use different images for training and testing data, the best way is to standardize them, and then covert to numbers. Thankfully, there is an open data set, which has already done this for us.

The Las Vegas Strip Data Set, published at the Machine Learning Repository of the University of California, Irvine – is ready to use data set. In case you wish to use your data set, you can process your's using the following approach, before using code on it:

DataSet



	User country	Nr. reviews	Nr. hotel reviews	Helpful votes	Score	Period of stay	Traveler type	Pool	Gym	Tennis court	Spa	Casino	Free internet	Hotel name	Hotel stars	Nr. rooms	User continent	Member years	Review month	Review weekday
0	USA	11	4	13	5	Dec-Feb	Friends	NO	YES	NO	NO	YES	YES	Circus Circus Hotel & Casino Las Vegas	3	3773	North America	9	January	Thursday
1	USA	119	21	75	3	Dec-Feb	Business	NO	YES	NO	NO	YES	YES	Circus Circus Hotel & Casino Las Vegas	3	3773	North America	3	January	Friday
2	USA	36	9	25	5	Mar-May	Families	NO	YES	NO	NO	YES	YES	Circus Circus Hotel & Casino Las Vegas	3	3773	North America	2	February	Saturday
3	UK	14	7	14	4	Mar-	Friends	NO	YES	NO	NO	YES	YES	Circus Circus Hotel &	3	3773	Europe	6	February	Friday

Data Set Link- https://archive.ics.uci.edu/ml/machine-learning-databases/00397/

Importing Libraries

We will start by importing the libraries. I have primarily used Sklearn, Pandas, and Numpy, along with Matplotlib to render charts. In Sklearn, I have used train_test_split to split the data sets; metrics, confusion_matrix, and precision_recall_fscore_support to check the accuracy and other metrics; KNeighborsClassifier and tree to use the KNN, SVC and Random Forest classifiers respectively.

```
import matplotlib.pyplot as plt
import matplotlib
import numpy as np
import pandas as pd
from IPython.display import display
```

Description of data in file-

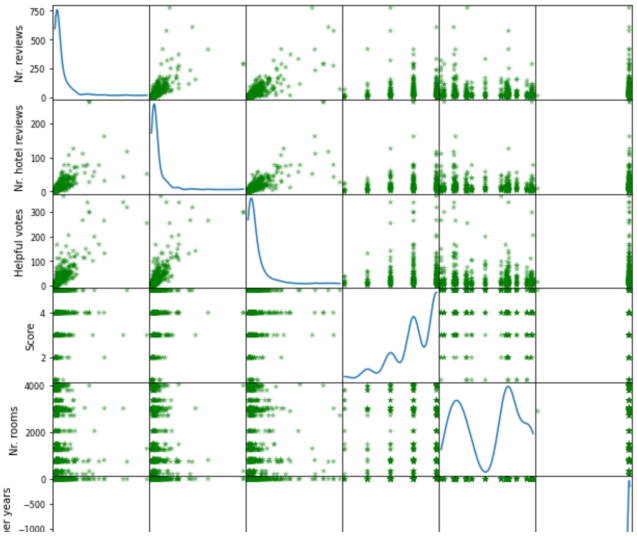
```
display(data.describe(include=[np.number]))
display(data.describe(exclude=[np.number]))
```

	Nr. reviews	Nr. hotel reviews	Helpful votes	Score	Nr. rooms	Member years
count	504.000000	504.000000	504.000000	504.000000	504.000000	504.000000
mean	48.130952	16.023810	31.751984	4.123016	2196.380952	0.767857
std	74.996426	23.957953	48.520783	1.007302	1285.476807	80.692897
min	1.000000	0.000000	0.000000	1.000000	188.000000	-1806.000000
25%	12.000000	5.000000	8.000000	4.000000	826.000000	2.000000
50%	23.500000	9.000000	16.000000	4.000000	2700.000000	4.000000
75%	54.250000	18.000000	35.000000	5.000000	3025.000000	6.000000
max	775.000000	263.000000	365.000000	5.000000	4027.000000	13.000000

	User country	Period of stay	Traveler type	Pool	Gym	Tennis court	Spa	Casino	Free internet	Hotel name	Hotel stars	User continent	Review month	Review weekday
count	504	504	504	504	504	504	504	504	504	504	504	504	504	504
unique	48	4	5	2	2	2	2	2	2	21	5	6	12	7
top	USA	Mar-May	Couples	YES	YES	NO	YES	YES	YES	The Venetian Las Vegas Hotel	5	North America	August	Wednesday
freq	217	128	214	480	480	384	384	456	480	24	192	295	42	85

Plotting scattered graph between the features-

```
pd.plotting.scatter_matrix(data,
                           figsize=(10,10),
                           diagonal='kde',
                           s=40,
                           alpha=0.5,
                           marker='*',
                           color='green');
```



Data Preprocessing-

```
le = preprocessing.LabelEncoder()

# seasons in place of months

['Dec-Feb' 'Mar-May' 'Jun-Aug' 'Sep-Nov']
data['Period of stay'] = data['Period of stay'].map({'Dec-Feb':'winter', 'Mar-May':'spring', 'Jun-Aug' :'summer','Sep-Nov':'autumn'})

for i in range(0, len(categorical)):
  #  data[categorical[i]] = le.fit_transform(data[categorical[i]])
    print(data[categorical[i]].unique())
```

Recursive feature elimination (RFE) to select features by recursively considering smaller and smaller sets of features-

```
##Applying random forest classifier on features to class

## Using feature selection pipeline and classification

from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import RFE
from sklearn.linear_model import Ridge


rfe = RFE(estimator = Ridge(), n_features_to_select = 12)
rfe.fit(X_train, y_train)
feature_list = pd.DataFrame({'col':list(X_train.columns.values),'sel':list(rfe.support_ *1)})
print("*Most contributing features in Score*")
cols=feature_list[feature_list.sel==1].col.values
print(feature_list[feature_list.sel==1].col.values)
cols=feature_list[feature_list.sel==1].col.values
```

```
*Most contributing features in Score*
['Period of stay' 'Traveler type' 'Pool' 'Gym' 'Spa' 'Casino'
 'Free internet' 'Hotel stars' 'User continent' 'Member years'
 'Review month' 'Review weekday']
```

Using Classifiers

In this project I have used three classifier to predict a suitable output with best accuracy score.

Random Forest Classsifier-

Using Random forest

```
[ ] clf = RandomForestClassifier(max_depth=5, random_state=0)
    clf.fit(X_sel, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=5, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=0, verbose=0,
                       warm_start=False)
```

Support Vector Classifier-

Using Support Vector Classifier

```
[ ] from sklearn.svm import SVC
    clf2=SVC()
    clf2.fit(X_sel,y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

KNN-

Using KNN for score prediction

```
[ ] from sklearn.neighbors import KNeighborsClassifier
    knn = KNeighborsClassifier(n_neighbors=10)
    knn.fit(X_sel, y_train)
```

By setting the data in all three classifier I have calculated the best score out of three classifier and used that classifier for further use.

```
print(knn.score(X_sel_t, y_test))
print(clf.score(X_sel_t, y_test))
print(clf2.score(X_sel_t,y_test))
```

```
0.42105263157894735
0.5
0.45394736842105265
```

Results-

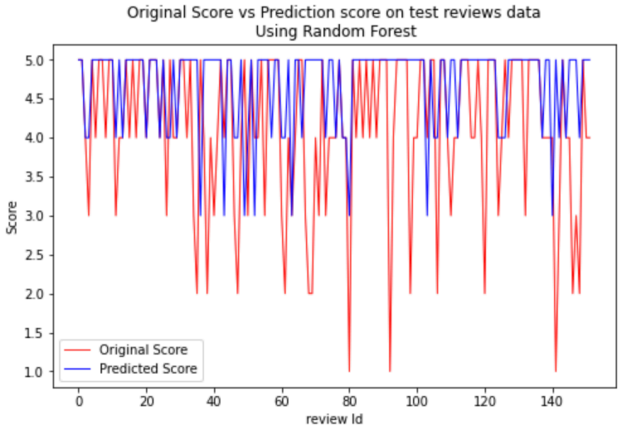
Plotting A table Between Original Score vs Prediction score on test reviews data Using Random Forest

```
[ ] Predictions = X_sel_t
    Predictions['Original_Score']= y_test
    Predictions['pred_score'] = p
    Predictions.head()
```

	Period of stay	Traveler type	Pool	Gym	Spa	Casino	Free internet	Hotel stars	User continent	Member years	Review month	Review weekday	Original_Score	pred_score
173	1	2	1	1	1	1	1	4	3	8	7	0	5	5
274	2	1	1	1	1	1	1	4	2	2	6	2	5	5
489	1	1	1	1	1	1	1	2	3	3	8	5	4	4
72	3	0	1	1	1	1	1	2	2	6	4	3	3	4
305	0	4	1	1	1	1	1	1	3	2	11	6	5	5

A prediction graph predicting the Original score vs per Predicted score by applying suitable features.

```
[ ] plt.figure(figsize=(8, 5))
    ax = plt.subplot()
    d = list(range(0,len(Predictions)))
    p1 = plt.plot(d,Predictions['Original_Score'],'r-', label="Original Score", linewidth= 1 )#,data['gamma'],data['test_err'],'b-')
    p2 = plt.plot(d,Predictions['pred_score'],'b-', label="Predicted Score", linewidth= 1)
    ax.set_title('Original Score vs Prediction score on test reviews data\n Using Random Forest')
    handles, labels = ax.get_legend_handles_labels()
    ax.legend(handles, labels)
    ax.set_xlabel('review Id')
    ax.set_ylabel('Score')
    plt.show()
```



Summary-

The best prediction score achived using Random Forest with 12 features out of 19, is 46% which is fairly low as the data is not so rich in my opinion. However Random Forest performed better than KNN in score prediction

References-

https://archive.ics.uci.edu/ml/machine-learning-databases/00397/

https://data.mendeley.com/datasets/tsf9sjdwh2

https://rdrr.io/cran/fcaR/man/vegas.html

https://www.academia.edu/34070065/Stripping_customers_feedback_on_hotels_through_data_mining_the_case_of_Las_Vegas_Strip

Importing and Training Data Set

I imported the data set as a Pandas dataframe. The data set had some rows of data of different objects and 20 columns.

Prepare train and test sets

```
[ ]  ## prepare train and test labels

     from sklearn.model_selection import train_test_split
     X= data.drop(['Score'], axis=1) ## remove score label from data
     y = data['Score']
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```