TRAFFIC ANALYZE

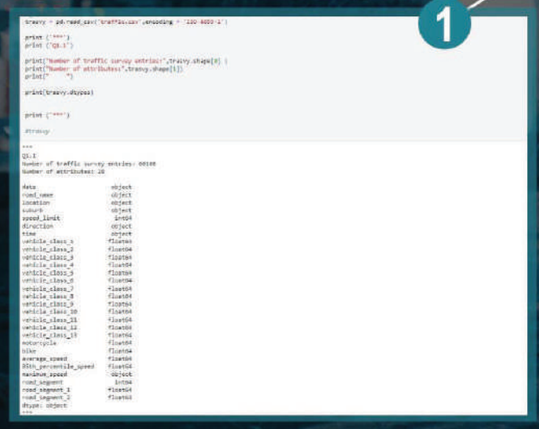
Data Science Project Development

2

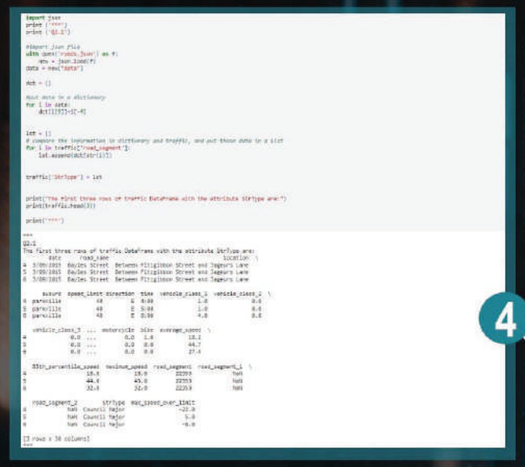
Introduction

Road fatalities in Victoria in the first half of 2019 increased by 50% compared with the same period in 2018. Across Australia, speeding was the single largest factor contributing to road fatalities. Over one third of these road fatalities occurred in capital cities. As a Data scientist, for this project, I applied various wrangling skills including processing and visualisation techniques to make sense of the data.

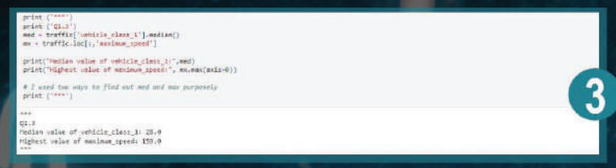
Print the number of traffic survey entries, number of attributes, attribute names and their data types from traffic.csv dataset. ▮



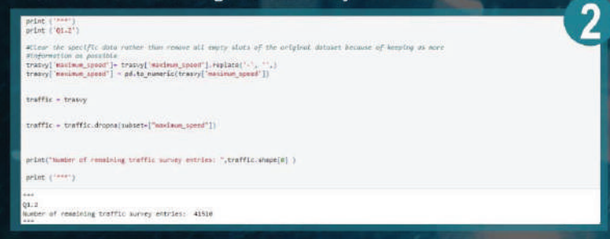
Using the traffic Data Frame as well as roads.json, add the attribute StrType from the JSON dataset to the traffic Data Frame. Print the first 3 rows. ▮



The vehicle class 1 attribute represents the number of short vehicles, such as cars, detected in a survey hour. Through traffic Data Frame, the median value of vehicle class 1 and highest maximum speed detected across all survey entries were calculated. ▮

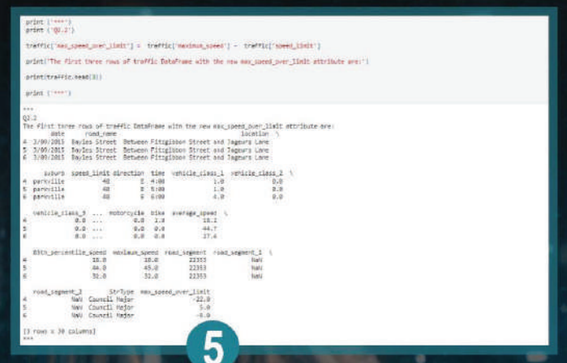


A number of survey entries detected no vehicles of any type. For some of these entries, a maximum speed of '-' has been included. For others, the maximum speed is blank. These entries can cause problems when analysing data. Create a Data Frame traffic from traffic.csv dataset with all such entries removed and print the number of remaining traffic survey entries. ▮

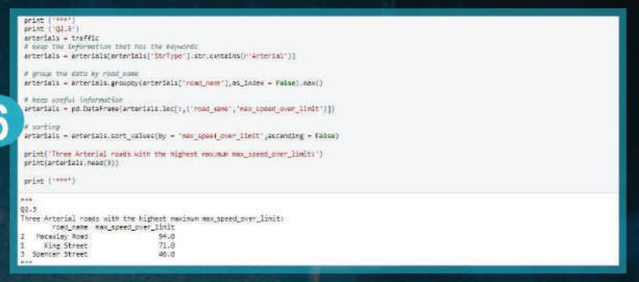


Perform K-means clustering on all data in the special traffic Data Frame using all at tributes except for StrType and idx as features. The SSE (sum of squared errors) can be used to measure the quality of clustering, which is the sum of distances of objects from their cluster centroids. Also, use the 'elbow method' to identify the optimal value of k from the plot. ▮

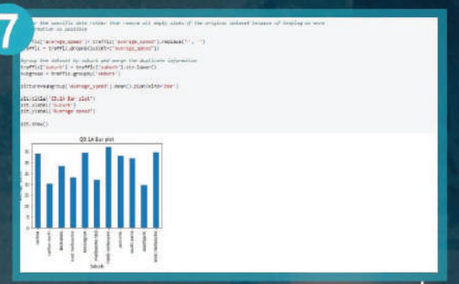
Add a new attribute max speed over limit to traffic Data Frame. This column will represent the difference between the maximum speed and the speed limit attributes; that is: max speed over limit = maximum speed- speed limit. Print the first 3 rows. ▮

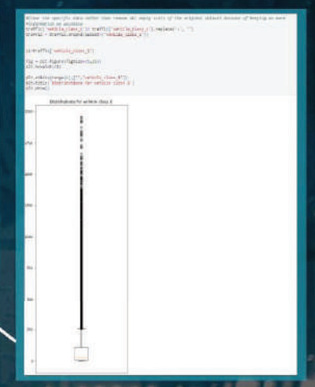


We are particularly concerned with instances of speeding on arterial roads, as arterial roads are the responsibility of VicRoads rather than local councils. Create a new Data Frame, arterials, which contains only the survey entries from traffic that relate to Arterial roads. Group the survey results in the arterials Data Frame byroad name. Print the names of the three roads with the highest maximum max speed over limit. ▮

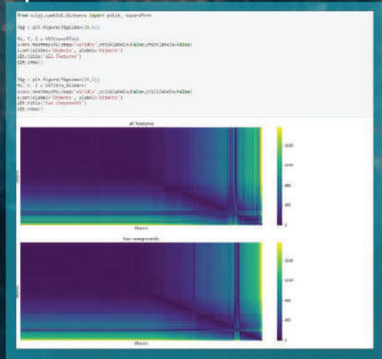


Using the traffic Data Frame, draw a bar plot showing suburb (x-axis) versus mean average speed (y-axis) and a Tukey boxplot showing the distribution of vehicle class 1 (the number of short vehicles) within the traffic survey results. ▮





The special traffic.csv dataset contains entries from traffic.csv with some added information. Although there are fewer entries than in the original dataset, there are still too many to visualise efficiently. For this set of visual analysis, we wish to create a Data Frame special traffic by randomly sampling 1000 entries. Using this special traffic Data Frame and all attributes except for StrType and idx as features to perform Principal Component Analysis and two VAT visual analyses. ▮



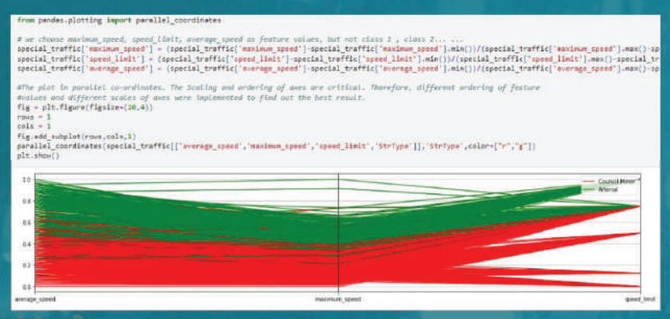
FINDINGS AND RECOMMENDATIONS

CODE: github.com/chenjiang0819/TrafficAnalyse

Evaluation 1

We can use Parallel Co-ordinates plot to evaluate K-means model against the measurement statement visually. Parallel Co-ordinates is a method for data visualisation. Each data instance is represented by a line and each feature by a vertical bar. The similarity of their lines can identify similar objects as well as correlations between (adjacent) features. Often, the lines of parallel co-ordinates plot representing a distinct class of objects group together, at least for some features.

```
from pandas.plotting import parallel_coordinates

# we choose maximum_speed, speed_limit, average_speed as feature values, but not class 1 , class 2... ...
special_traffic['maximum_speed'] = (special_traffic['maximum_speed']-special_traffic['maximum_speed'].min())/(special_traffic['maximum_speed'].max()-sp
special_traffic['speed_limit'] = (special_traffic['speed_limit']-special_traffic['speed_limit'].min())/(special_traffic['speed_limit'].max()-special_tr
special_traffic['average_speed'] = (special_traffic['average_speed']-special_traffic['average_speed'].min())/(special_traffic['average_speed'].max()-sp

#The plot in parallel co-ordinates. The Scaling and ordering of axes are critical. Therefore, different ordering of feature
#values and different scales of axes were implemented to find out the best result.
fig = plt.figure(figsize=(20,4))
rows = 1
cols = 1
fig.add_subplot(rows,cols,1)
parallel_coordinates(special_traffic[['average_speed','maximum_speed','speed_limit','StrType']],'StrType',color=["r","g"])
plt.show()
```



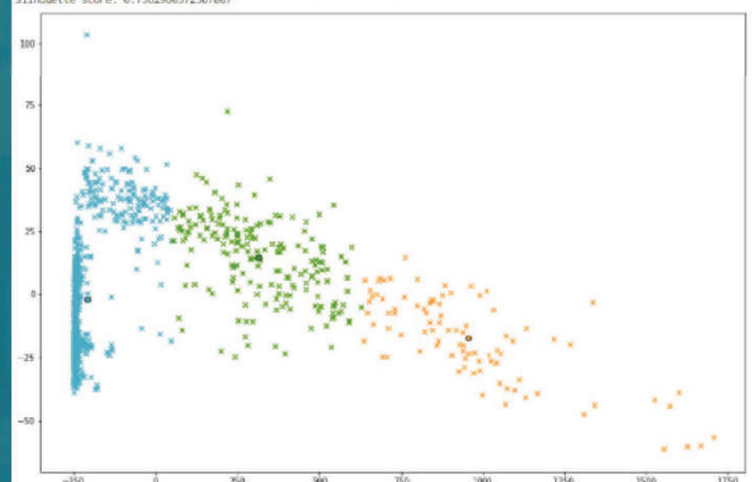
We can see that there are five vertical bar groups at the end, and converged into three different directions. It determined that the K-means method is a proper tool to classify the data set in this case.

Result

We can plot a scatter graph to evaluate Kmeans model because we are able to identify the group of data points and the centre of clusters. Therefore, we can assess the kmean model with bare eyes.

```
pre=kmeans.predict(tra_sklearn)

samples,features=tra_sklearn.shape
inertias=kmeans.inertia_


silhouette_s=metrics.silhouette_score(tra_sklearn,pre,metric='euclidean')

print("Total samples:",samples)
print("Total features:",features)
print("Sum of squared distances of samples to their closest cluster center:",inertias)
print("Silhouette score:",silhouette_s)



#Plot the graph of
centers= kmeans.cluster_centers_
# setting the colours
colors=['#4EACC5','#FF9C34','#4E9A06']
plt.figure(figsize=(15,10))
for i in range(3):
    index_sets=np.where(pre==i)
    cluster=tra_sklearn[index_sets]

    #show the all data in the clusters
    plt.scatter(cluster[:,0],cluster[:,1],c=colors[i],marker='x')

    #show the centroids of each cluster
    plt.plot(centers[i][0],centers[i][1],'o',markerfacecolor=colors[i]
             ,markeredgecolor='k',markersize=6)
plt.show()
```

```
Total samples: 1000
Total features: 2
Sum of squared distances of samples to their closest cluster center: 14512024.78119726
Silhouette score: 0.7562900372507007
```

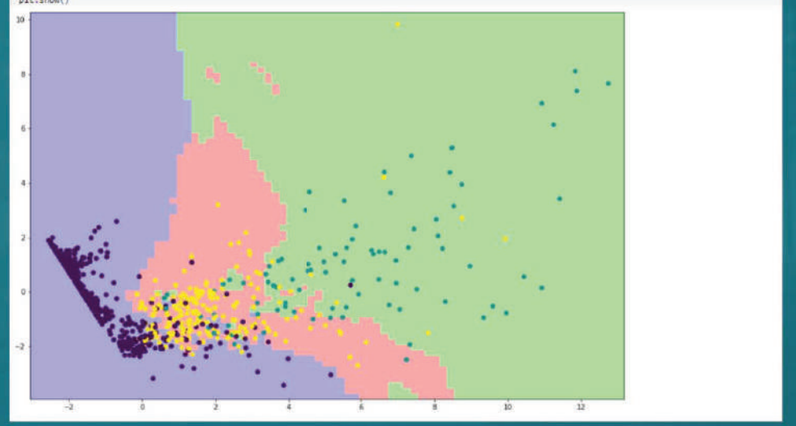


After plotting the scatter graph, we can identify that the clusters are quite reasonable. Also, the Silhouette score (0.7475097984679786) is relatively high.

Evaluation 2

We also can use the K-Nearest Neighbors(KNN) method to evaluate the result of K-mean.

```
from sklearn.neighbors import KNeighborsClassifier
from matplotlib.colors import ListedColormap

tra_sklearn1 = special_traffic

# wash data first
tra_sklearn1= tra_sklearn1.replace('-', '')
tra_sklearn1 = tra_sklearn1.dropna()

tra_sklearn1 = tra_sklearn1.iloc[:,1:19]

#standardisation
scaler = StandardScaler().fit(tra_sklearn1.values)
tra_sklearn1 = scaler.transform(tra_sklearn1.values)


#we want just the first two PCs
sklearn_pca = sklearnPCA(n_components=2)
tra_sklearn2 = sklearn_pca.fit_transform(tra_sklearn1)



indexx = []


x_min, x_max = tra_sklearn2[:,0].min() - 0.5, tra_sklearn2[:,0].max() + 0.5
y_min, y_max = tra_sklearn2[:,1].min() - 0.5, tra_sklearn2[:,1].max() + 0.5


cmap_light = ListedColormap(['#AAAAFF','#AAFFAA','#FFAAAA'])

h = 0.2


xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max,h))

for i in range(tra_sklearn2.shape[0]):
    indexx.append(i)

# implement the knn method on data set

knn = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                           metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                           weights='uniform')

knn.fit(tra_sklearn2,label)


result = knn.predict(np.c_[xx.ravel(),yy.ravel()])
result = result.reshape(xx.shape)

plt.figure(figsize=(15,10))
plt.pcolormesh(xx,yy,result,cmap = cmap_light)


# Plot the training points
plt.scatter(tra_sklearn2[:,0],tra_sklearn2[:,1],c = label)
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.show()
```



We can see that the result of the KNN method is similar to that of the Kmean method. Therefore, we can confirm that the outcome of Kmeans is at a global optimal.