ROAD ACCIDENT ANALYSIS

A MINI PROJECT REPORT SUBMITTED BY

Chinthan B	Emmanuel Joshy
4NM20CS056	4NM20CS069
VI Semester, B Section	VI Semester, B section

UNDER THE GUIDANCE OF

Ms. Vaishali Bangera
Assistant Professor GD I
Department of Computer Science and Engineering

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Bachelor of Engineering in Computer Science & Engineering

from

Visvesvaraya Technological University, Belagavi

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum)
AICTE approved, (ISO 9001:2015 Certified), Accredited with ‘A’ Grade by NAAC
NITTE -574 110, Udupi District, KARNATAKA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

May 2023

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

CERTIFICATE

"Road Accident Analysis" is a Bonafide work carried out by Emmanuel Joshy (4NM20CS069) and Chinthan B (4NM20CS056) in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering prescribed by Visvesvaraya Technological University, Belagavi during the year 2022-2023. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report. The Mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide   Signature of HOD

ABSTRACT

This study examines the relationship between road accidents and various factors such as lighting conditions, road surface conditions, junction control, attention from the police, and urban or rural area type. The data were collected from various sources and analyzed using statistical methods to identify significant trends and patterns. The results suggest that poor lighting conditions, wet or slippery road surfaces, inadequate junction control, and urban areas with high traffic volume are all significant factors in road accidents.

Here we use a dataset containing details of various accidents that took place in United Kingdom from 2017-2021. We perform data preprocessing to enhance the models accuracy and performance. We evaluate the performance of the model using random forest algorithm and then calculate the accuracy score, Precision score, recall and F1-score.

The findings of this analysis provide insights into the relationship between light conditions and road accidents, highlighting the potential for accurate prediction based on other available variables. These findings can inform road safety measures and aid in developing strategies to reduce accidents based on light conditions. It suggest that lighting conditions and road surface conditions are important factors contributing to road accidents. The analysis also shows that the presence of junction controls and attention from the police can have a positive impact on reducing the number of accidents.

Additionally, the study highlights the differences in accident rates between urban and rural areas, indicating a need for tailored interventions to improve road safety in these areas. Overall, this study provides valuable insights into the factors affecting road safety and can inform the policy and the intervention efforts aimed at reducing the number of road accidents caused.

Table of Contents

INTRODUCTION

Road accidents are a major cause of concern across the world. They result in loss of life, injury, and significant economic costs. Road accidents can occur due to various reasons, including driver error, vehicle malfunction, and poor road infrastructure. In order to understand the causes of road accidents, it is important to analyze various factors that contribute to them.

One of the important factors that can contribute to road accidents is lighting conditions. Poor lighting conditions can make it difficult for drivers to see the road ahead, especially at night. Similarly, road surface conditions, such as wet or slippery surfaces, can also contribute to accidents, particularly in adverse weather conditions.

Junction control is another important factor that can contribute to road accidents. Poorly designed junctions or inadequate traffic control measures can lead to confusion among drivers and increase the risk of accidents.

The attention from the police is also an important factor in road safety. Enforcement of traffic laws, including speeding and drunk driving, can help reduce the number of accidents on the road.

Finally, the type of area where the accident occurs can also play a significant role. Urban areas are more likely to have higher traffic volumes and more complex road systems, which can increase the risk of accidents. On the other hand, rural areas may have higher speeds and more hazardous road conditions, such as sharp bends and steep inclines. Overall, a comprehensive analysis of these factors can provide valuable insights into the causes of road accidents and help identify strategies to reduce their frequency and severity.

SOFTWARE REQUIREMENTS

- Operating System: Windows 11
- Jupyter notebook: The machine learning model was built using Jupyter Notebook, an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text.
- Pandas: It is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool that allows data scientists to work with structured data efficiently. The library was used to preprocess and clean the dataset, including feature selection, data normalization and data transformation.
- NumPy: It is a fundamental package for scientific computing in Python, providing support for large, multi-dimensional arrays and matrices, along with a vast library of mathematical functions. NumPy was used to create the array of features and labels and for model evaluation.
- Seaborn: It is a Python data visualization library based on Matplotlib that provides a high-level interface for drawing informative and attractive statistical graphics. Seaborn was used to create various visualizations, including scatterplots, boxplots, and histograms, to explore the data and gain insights.
- Matplotlib: It is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib was used to create the final visualizations of the model's performance.

HARDWARE REQUIREMENTS

- RAM: Minimum 8GB (8GB recommended)
- Processor: Intel i5 or above
- HDD: 20 GB Disk space

LITERATURE SURVEY

1. Adams, J., Hill, D., & Yau, K. (2017). Investigation of factors contributing to serious road accidents. Accident Analysis & Prevention, 98, 241-249

   Objective: This study aimed to investigate the factors contributing to serious road accidents and analyze their impact on accident occurrence and severity.

   Findings: The study found that factors such as driver age, road type, weather conditions, and light conditions significantly influenced the occurrence and severity of road accidents.

2. Quddus, M. A., Ochieng, W. Y., & Noland, R. B. (2007). Current and future trends in UK road accidents and casualties. Accident Analysis & Prevention, 39(3), 611-624.

   Objective: This study aimed to analyze current and future trends in road accidents and casualties in the UK and identify the factors influencing accident rates.

   Findings: The study identified light conditions as a significant factor affecting accident rates, with higher accident rates observed during low light and dark conditions. The findings also emphasized the importance of considering temporal trends and demographic factors in accident analysis.

3. Chen, C., & Zhang, G. (2019). Analysis of road accident severity using machine learning algorithms. Journal of Safety Research, 70, 11-22.

   Objective: This study aimed to analyze road accident severity using machine learning algorithms and identify the key factors contributing to accident severity.

   Findings: The study demonstrated that machine learning models can effectively predict accident severity based on factors such as light conditions, road type, weather conditions.

DESIGN AND ANALYSIS

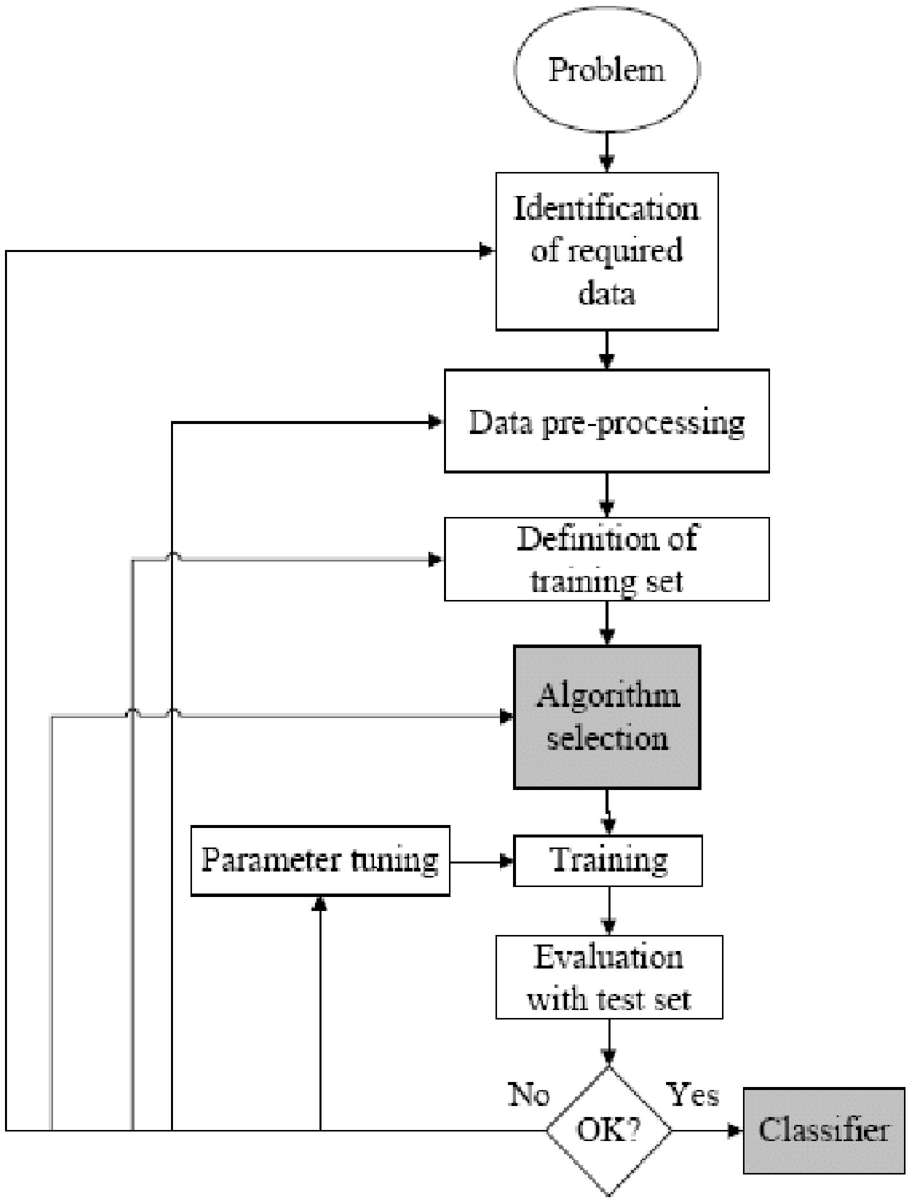


Fig: Flow diagram of project

The above flow chart shows the overall structure of the project. Initially we need to setup all the requirements both software and hardware. Then dataset is given as input to the system and then process goes as per the flow diagram.

Problem Identification: To analyse road accidents based on certain conditions.

Identification of Required Data: Determine the specific data required to perform the analysis. This may include variables such as accident date and time, location, weather conditions, light conditions, road type, and any other relevant factors. Ensure that you have access to a reliable and comprehensive dataset that includes these variables.

Data Preprocessing: Clean and preprocess the raw data to ensure its quality and usability. This step may involve handling missing values, removing duplicates, correcting inconsistencies, and transforming variables into appropriate formats.

Definition of Training Set: Split the dataset into a training set and a test set. The training set will be used to train the machine learning model, while the test set will be used to evaluate its performance.

Algorithm Selection: Select an appropriate machine learning algorithm or a combination of algorithms for your analysis. Consider algorithms such as decision trees, random forests, logistic regression, or support vector machines, depending on the nature of your problem and the available data.

Training: Train with Random Forest algorithm using the training set. This involves feeding the algorithm with the labeled data, allowing it to learn patterns and relationships between the input features (e.g., road surface conditions, light conditions) and the target variable (e.g., accident severity).

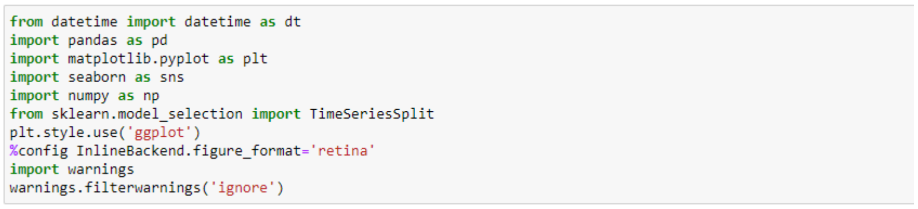
Parameter Tuning: Adjust the values of hyperparameters associated with the chosen algorithm(s) to find the best combination for optimal model performance. This can be done through techniques like grid search, random search, or Bayesian optimization. Iterate through different parameter values and evaluate the model's performance on a validation set.

Evaluation with Test Set: Apply the trained model to the test set to assess its performance. Evaluate metrics such as accuracy, precision, recall, and F1 score to measure the effectiveness of the model in predicting accident outcomes based on the given features.

Classifier: Based on the evaluation results, determine the effectiveness of the chosen algorithm as a classifier for road accidents. If the performance meets your desired criteria, you can consider the trained model as a reliable classifier for predicting accident outcomes based on the input variables.

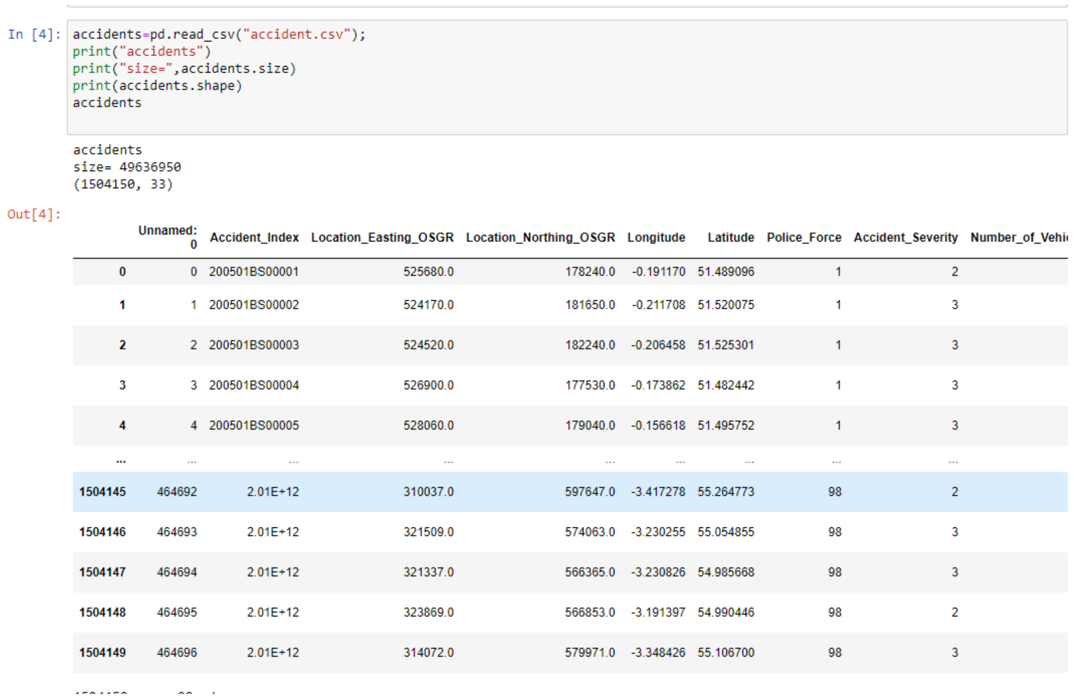
IMPLEMENTATION

- The following Libraries are to be installed and imported to run the project:



```
from datetime import datetime as dt
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.model_selection import TimeSeriesSplit
plt.style.use('ggplot')
%config InlineBackend.figure_format='retina'
import warnings
warnings.filterwarnings('ignore')
```

- The required dataset is being loaded and printed:

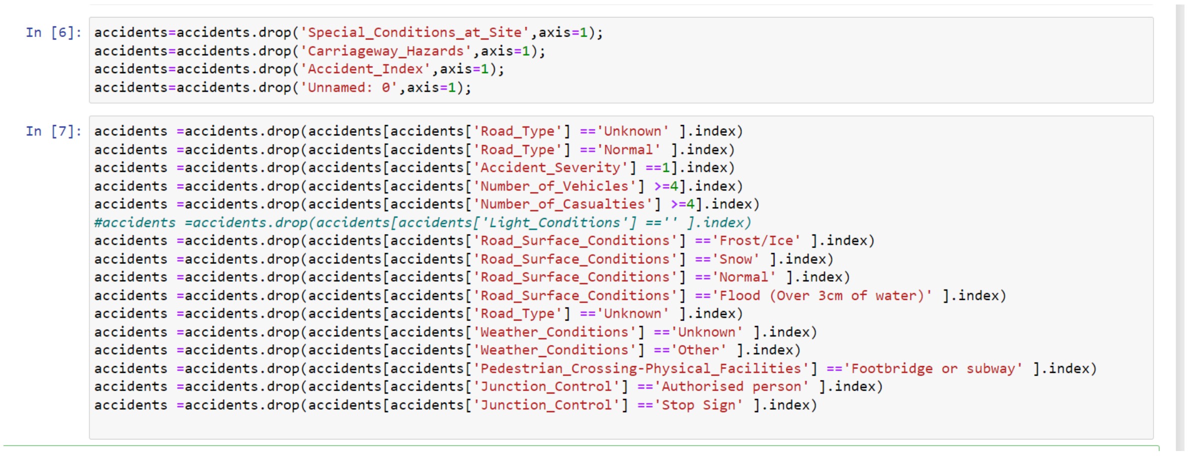


```
In [4]: accidents=pd.read_csv("accident.csv");
        print("accidents")
        print("size=",accidents.size)
        print(accidents.shape)
        accidents

accidents
size= 49636950
(1504150, 33)
```

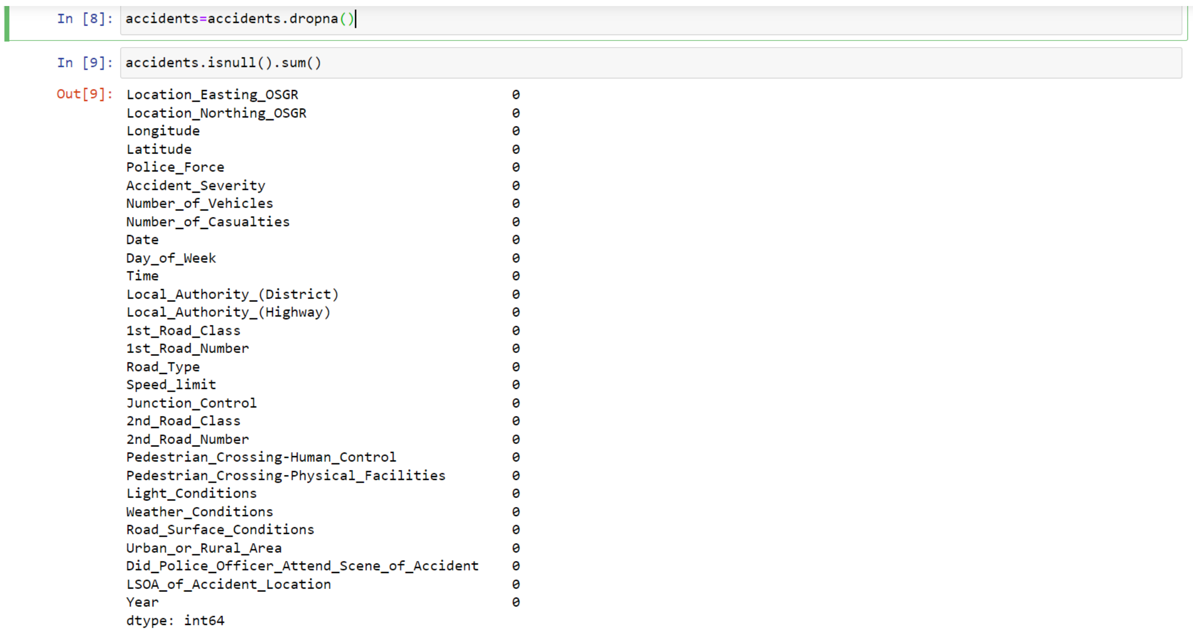
Out[4]:

	Unnamed: 0	Accident_Index	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	Latitude	Police_Force	Accident_Severity	Number_of_Vehi
0	0	200501BS00001	525680.0	178240.0	-0.191170	51.489096	1	2	
1	1	200501BS00002	524170.0	181650.0	-0.211708	51.520075	1	3	
2	2	200501BS00003	524520.0	182240.0	-0.206458	51.525301	1	3	
3	3	200501BS00004	526900.0	177530.0	-0.173862	51.482442	1	3	
4	4	200501BS00005	528060.0	179040.0	-0.156618	51.495752	1	3	
...	...	...	...	...	...	...	...	...	
1504145	464692	2.01E+12	310037.0	597647.0	-3.417278	55.264773	98	2	
1504146	464693	2.01E+12	321509.0	574063.0	-3.230255	55.054855	98	3	
1504147	464694	2.01E+12	321337.0	566365.0	-3.230826	54.985668	98	3	
1504148	464695	2.01E+12	323869.0	566853.0	-3.191397	54.990446	98	2	
1504149	464696	2.01E+12	314072.0	579971.0	-3.348426	55.106700	98	3	

- Removing null values and data with missing values and checking for null value

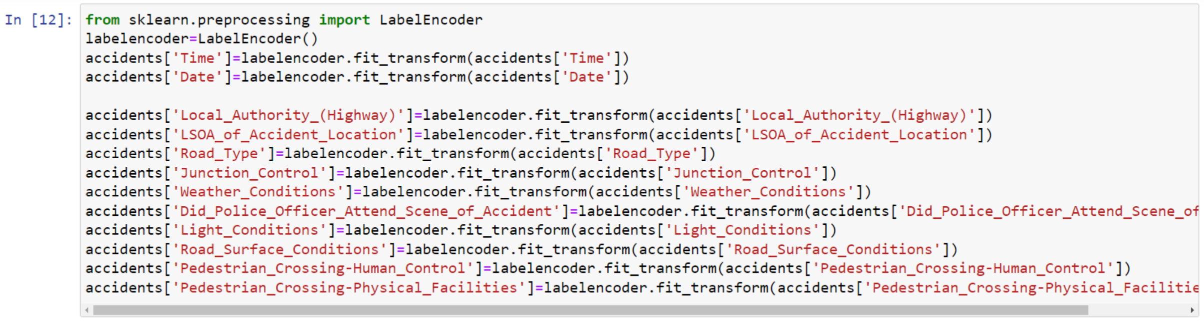


```
In [6]: accidents=accidents.drop('Special_Conditions_at_Site',axis=1);
        accidents=accidents.drop('Carriageway_Hazards',axis=1);
        accidents=accidents.drop('Accident_Index',axis=1);
        accidents=accidents.drop('Unnamed: 0',axis=1);

In [7]: accidents =accidents.drop(accidents[accidents['Road_Type'] =='Unknown' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Type'] =='Normal' ].index)
        accidents =accidents.drop(accidents[accidents['Accident_Severity'] ==1].index)
        accidents =accidents.drop(accidents[accidents['Number_of_Vehicles'] >=4].index)
        accidents =accidents.drop(accidents[accidents['Number_of_Casualties'] >=4].index)
        #accidents =accidents.drop(accidents[accidents['Light_Conditions'] =='' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Surface_Conditions'] =='Frost/Ice' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Surface_Conditions'] =='Snow' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Surface_Conditions'] =='Normal' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Surface_Conditions'] =='Flood (Over 3cm of water)' ].index)
        accidents =accidents.drop(accidents[accidents['Road_Type'] =='Unknown' ].index)
        accidents =accidents.drop(accidents[accidents['Weather_Conditions'] =='Unknown' ].index)
        accidents =accidents.drop(accidents[accidents['Weather_Conditions'] =='Other' ].index)
        accidents =accidents.drop(accidents[accidents['Pedestrian_Crossing-Physical_Facilities'] =='Footbridge or subway' ].index)
        accidents =accidents.drop(accidents[accidents['Junction_Control'] =='Authorised person' ].index)
        accidents =accidents.drop(accidents[accidents['Junction_Control'] =='Stop Sign' ].index)
```



```
In [8]: accidents=accidents.dropna()

In [9]: accidents.isnull().sum()

Out[9]: Location_Easting_OSGR                          0
        Location_Northing_OSGR                         0
        Longitude                                      0
        Latitude                                       0
        Police_Force                                   0
        Accident_Severity                              0
        Number_of_Vehicles                             0
        Number_of_Casualties                           0
        Date                                           0
        Day_of_Week                                    0
        Time                                           0
        Local_Authority_(District)                     0
        Local_Authority_(Highway)                      0
        1st_Road_Class                                 0
        1st_Road_Number                                0
        Road_Type                                      0
        Speed_limit                                    0
        Junction_Control                               0
        2nd_Road_Class                                 0
        2nd_Road_Number                                0
        Pedestrian_Crossing-Human_Control              0
        Pedestrian_Crossing-Physical_Facilities        0
        Light_Conditions                               0
        Weather_Conditions                             0
        Road_Surface_Conditions                        0
        Urban_or_Rural_Area                            0
        Did_Police_Officer_Attend_Scene_of_Accident    0
        LSOA_of_Accident_Location                      0
        Year                                           0
        dtype: int64
```

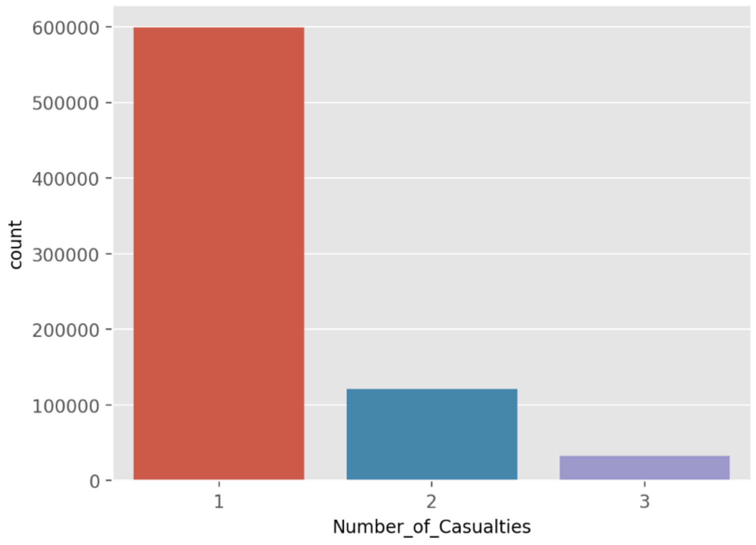
- Encode the string data in the dataset to numeric value:



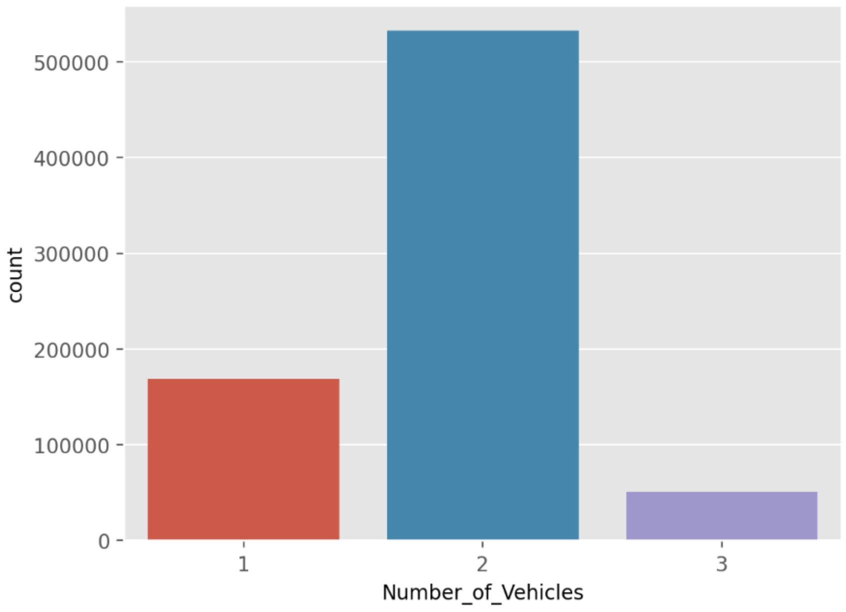
```
In [12]: from sklearn.preprocessing import LabelEncoder
         labelencoder=LabelEncoder()
         accidents['Time']=labelencoder.fit_transform(accidents['Time'])
         accidents['Date']=labelencoder.fit_transform(accidents['Date'])

         accidents['Local_Authority_(Highway)']=labelencoder.fit_transform(accidents['Local_Authority_(Highway)'])
         accidents['LSOA_of_Accident_Location']=labelencoder.fit_transform(accidents['LSOA_of_Accident_Location'])
         accidents['Road_Type']=labelencoder.fit_transform(accidents['Road_Type'])
         accidents['Junction_Control']=labelencoder.fit_transform(accidents['Junction_Control'])
         accidents['Weather_Conditions']=labelencoder.fit_transform(accidents['Weather_Conditions'])
         accidents['Did_Police_Officer_Attend_Scene_of_Accident']=labelencoder.fit_transform(accidents['Did_Police_Officer_Attend_Scene_of
         accidents['Light_Conditions']=labelencoder.fit_transform(accidents['Light_Conditions'])
         accidents['Road_Surface_Conditions']=labelencoder.fit_transform(accidents['Road_Surface_Conditions'])
         accidents['Pedestrian_Crossing-Human_Control']=labelencoder.fit_transform(accidents['Pedestrian_Crossing-Human_Control'])
         accidents['Pedestrian_Crossing-Physical_Facilities']=labelencoder.fit_transform(accidents['Pedestrian_Crossing-Physical_Facilitie
```

- Analysis:

```
In [21]: sns.countplot(x=accidents['Number_of_Casualties'])
Out[21]: <Axes: xlabel='Number_of_Casualties', ylabel='count'>
```

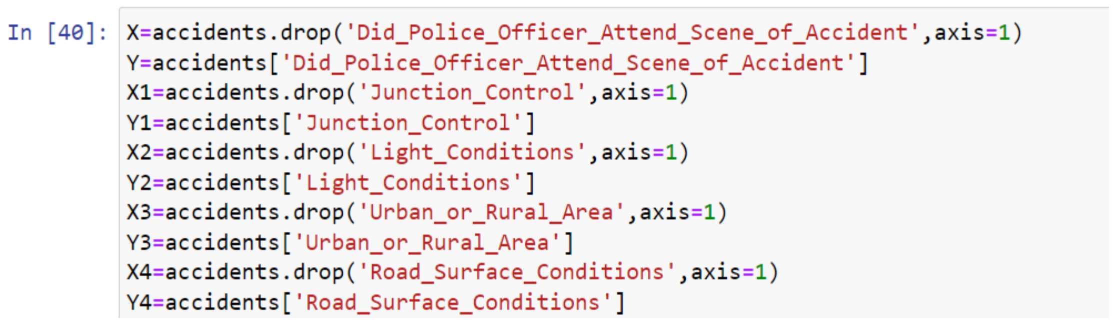


```
In [20]: sns.countplot(x=accidents['Number_of_Vehicles'])
Out[20]: <Axes: xlabel='Number_of_Vehicles', ylabel='count'>
```



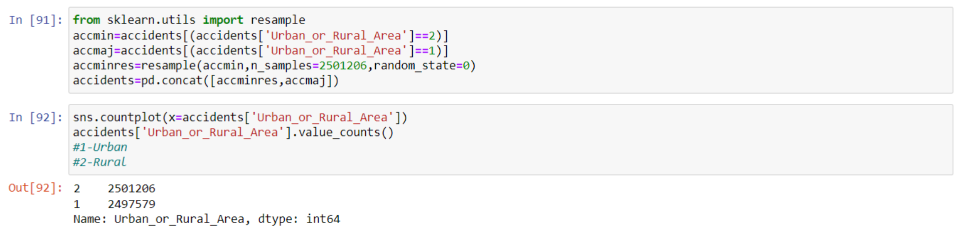
- ## Feature Importance :

  Classification version of target variable:

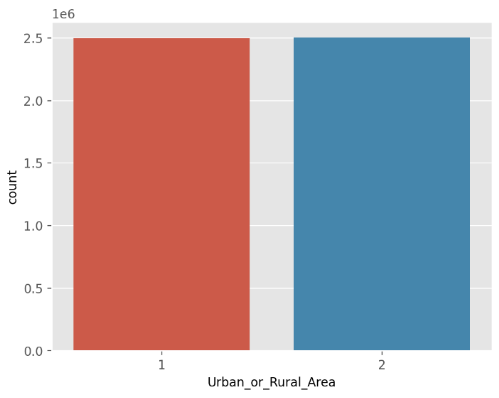


```
In [40]: X=accidents.drop('Did_Police_Officer_Attend_Scene_of_Accident',axis=1)
         Y=accidents['Did_Police_Officer_Attend_Scene_of_Accident']
         X1=accidents.drop('Junction_Control',axis=1)
         Y1=accidents['Junction_Control']
         X2=accidents.drop('Light_Conditions',axis=1)
         Y2=accidents['Light_Conditions']
         X3=accidents.drop('Urban_or_Rural_Area',axis=1)
         Y3=accidents['Urban_or_Rural_Area']
         X4=accidents.drop('Road_Surface_Conditions',axis=1)
         Y4=accidents['Road_Surface_Conditions']
```

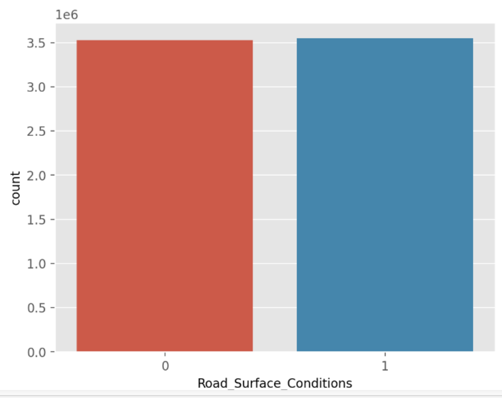
- ## Resampling of data for better accuracy:



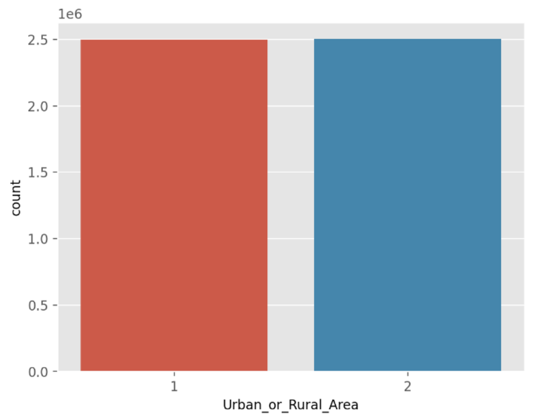
```
In [91]: from sklearn.utils import resample
         accmin=accidents[(accidents['Urban_or_Rural_Area']==2)]
         accmaj=accidents[(accidents['Urban_or_Rural_Area']==1)]
         accminres=resample(accmin,n_samples=2501206,random_state=0)
         accidents=pd.concat([accminres,accmaj])

In [92]: sns.countplot(x=accidents['Urban_or_Rural_Area'])
         accidents['Urban_or_Rural_Area'].value_counts()
         #1-Urban
         #2-Rural

Out[92]: 2    2501206
         1    2497579
         Name: Urban_or_Rural_Area, dtype: int64
```



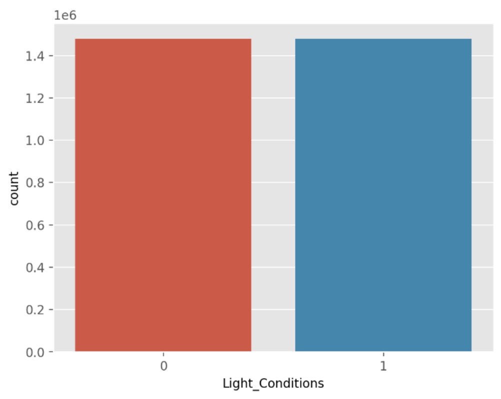
```
In [109]: from sklearn.utils import resample
          accmin=accidents[(accidents['Road_Surface_Conditions']==1)]
          accmaj=accidents[(accidents['Road_Surface_Conditions']==0)]
          accminres=resample(accmin,n_samples=3546783,random_state=0)
          accidents=pd.concat([accminres,accmaj])

In [110]: sns.countplot(x=accidents['Road_Surface_Conditions'])
          accidents['Road_Surface_Conditions'].value_counts()
          #0-Dry
          #1-Wet/Damp

Out[110]: 1    3546783
          0    3529144
          Name: Road_Surface_Conditions, dtype: int64
```



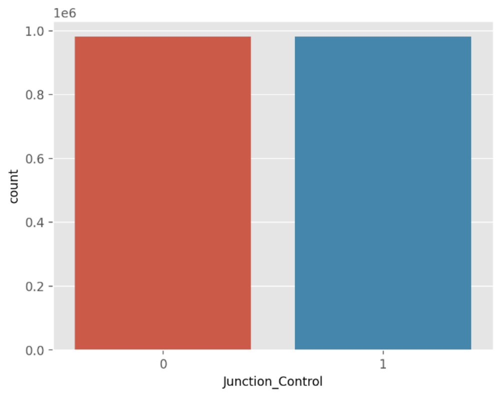
```
In [91]: from sklearn.utils import resample
         accmin=accidents[(accidents['Urban_or_Rural_Area']==2)]
         accmaj=accidents[(accidents['Urban_or_Rural_Area']==1)]
         accminres=resample(accmin,n_samples=2501206,random_state=0)
         accidents=pd.concat([accminres,accmaj])

In [92]: sns.countplot(x=accidents['Urban_or_Rural_Area'])
         accidents['Urban_or_Rural_Area'].value_counts()
         #1-Urban
         #2-Rural

Out[92]: 2    2501206
         1    2497579
         Name: Urban_or_Rural_Area, dtype: int64
```



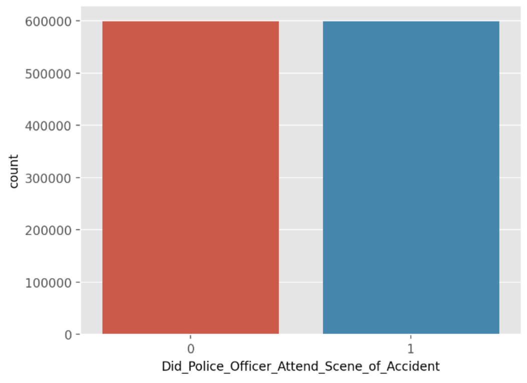
```
In [73]: from sklearn.utils import resample
         accmin=accidents[(accidents['Light_Conditions']==0)]
         accmaj=accidents[(accidents['Light_Conditions']==1)]
         accminres=resample(accmin,n_samples=1477997,random_state=0)
         accidents=pd.concat([accminres,accmaj])

In [74]: sns.countplot(x=accidents['Light_Conditions'])
         accidents['Light_Conditions'].value_counts()
         #0-Darkness
         #1-Daylight

Out[74]: 0    1477997
         1    1477997
         Name: Light_Conditions, dtype: int64
```



```
In [56]: from sklearn.utils import resample
         accmin=accidents[(accidents['Junction_Control']==0)]
         accmaj=accidents[(accidents['Junction_Control']==1)]
         accminres=resample(accmin,n_samples=980912,random_state=0)
         accidents=pd.concat([accminres,accmaj])

In [57]: sns.countplot(x=accidents['Junction_Control'])
         accidents['Junction_Control'].value_counts()
         #1-Automatic traffic Signal
         #2-Giveway or uncontrolled

Out[57]: 0    980912
         1    980912
         Name: Junction_Control, dtype: int64
```



```
In [38]: from sklearn.utils import resample
         accmin=accidents[(accidents['Did_Police_Officer_Attend_Scene_of_Accident']==0)]
         accmaj=accidents[(accidents['Did_Police_Officer_Attend_Scene_of_Accident']==1)]
         accminres=resample(accmin,n_samples=599045,random_state=0)
         accidents=pd.concat([accminres,accmaj])

In [39]: sns.countplot(x=accidents['Did_Police_Officer_Attend_Scene_of_Accident'])
         accidents['Did_Police_Officer_Attend_Scene_of_Accident'].value_counts()
         #0-no
         #1-yes

Out[39]: 0    599045
         1    599045
         Name: Did_Police_Officer_Attend_Scene_of_Accident, dtype: int64
```



- Splitting the Dataset:

```
In [111]: from sklearn.model_selection import train_test_split
          from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score
          X_train,X_test,y_train,y_test = train_test_split(X4,Y4,test_size=0.2,random_state=0)

In [93]: from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score
         X_train,X_test,y_train,y_test = train_test_split(X3,Y3,test_size=0.2,random_state=0)

In [75]: from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score
         X_train,X_test,y_train,y_test = train_test_split(X2,Y2,test_size=0.2,random_state=0)

In [58]: from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score
         X_train,X_test,y_train,y_test = train_test_split(X1,Y1,test_size=0.2,random_state=0)

In [41]: from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score
         X_train,X_test,y_train,y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
```

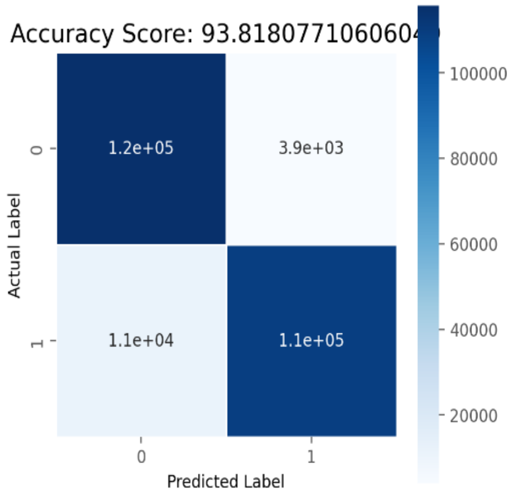
RESULTS

Random forest classifier:

1. Did Police Officer attend the accident:

```
In [43]:
Y_predict=rfc.predict(X_test)
print("Accuracy Score:",accuracy_score(y_test,Y_predict)*100,"%")
print("F1 Score:",f1_score(y_test,Y_predict))
print("Recall Score:",recall_score(y_test,Y_predict))
print("Precision Score:",precision_score(y_test,Y_predict))

Accuracy Score: 93.81807710606049 %
F1 Score: 0.9364682469902512
Recall Score: 0.9088661338661339
Precision Score: 0.965799414361415
```

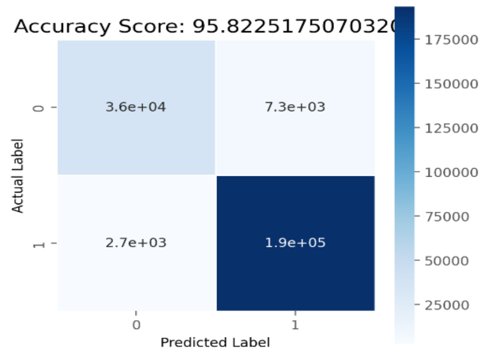
```
In [44]:
from sklearn.metrics import classification_report,confusion_matrix
cm=confusion_matrix(y_test,Y_predict)
plt.figure(figsize=(5,5))
sns.heatmap(data=cm,linewidths=0.5,annot=True,square=True,cmap='Blues')
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
all_sample_title = 'Accuracy Score: {0}'.format(rfc.score(X_test,y_test)*100)
plt.title(all_sample_title,size=15)

Out[44]: Text(0.5, 1.0, 'Accuracy Score: 93.81807710606049')
```



2. Junction control:

```
In [60]:
Y_predict=rfc.predict(X_test)
print("Accuracy Score:",accuracy_score(y_test,Y_predict)*100,"%")
print("F1 Score:",f1_score(y_test,Y_predict))
print("Recall Score:",recall_score(y_test,Y_predict))
print("Precision Score:",precision_score(y_test,Y_predict))

Accuracy Score: 95.82251750703202 %
F1 Score: 0.9747590637907296
Recall Score: 0.9860774540464153
Precision Score: 0.9636975544087951
```

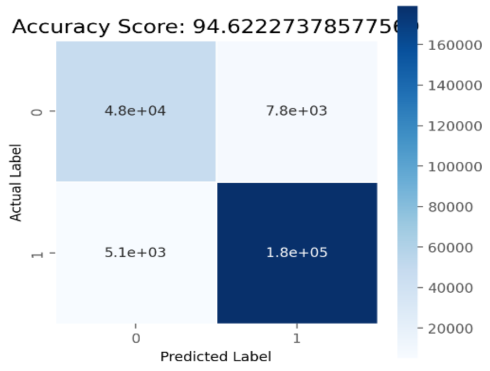
```
In [61]:
from sklearn.metrics import classification_report,confusion_matrix
cm=confusion_matrix(y_test,Y_predict)
plt.figure(figsize=(5,5))
sns.heatmap(data=cm,linewidths=0.5,annot=True,square=True,cmap='Blues')
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
all_sample_title = 'Accuracy Score: {0}'.format(rfc.score(X_test,y_test)*100)
plt.title(all_sample_title,size=15)
Out[61]: Text(0.5, 1.0, 'Accuracy Score: 95.82251750703202')
```



3. Light conditions:

```
In [77]:
Y_predict=rfc.predict(X_test)
print("Accuracy Score:",accuracy_score(y_test,Y_predict)*100,"%")
print("F1 Score:",f1_score(y_test,Y_predict))
print("Recall Score:",recall_score(y_test,Y_predict))
print("Precision Score:",precision_score(y_test,Y_predict))

Accuracy Score: 94.62227378577569 %
F1 Score: 0.965244550412394
Recall Score: 0.9722460689175533
Precision Score: 0.9583431521669273
```

```
In [78]:
from sklearn.metrics import classification_report,confusion_matrix
cm=confusion_matrix(y_test,Y_predict)
plt.figure(figsize=(5,5))
sns.heatmap(data=cm,linewidths=0.5,annot=True,square=True,cmap='Blues')
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
all_sample_title = 'Accuracy Score: {0}'.format(rfc.score(X_test,y_test)*100)
plt.title(all_sample_title,size=15)
Out[78]: Text(0.5, 1.0, 'Accuracy Score: 94.62227378577569')
```



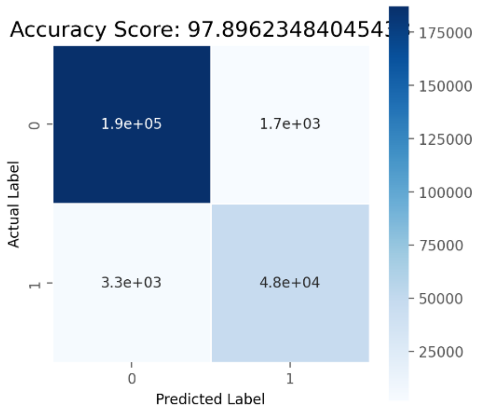
4.Urban or rural area:

In [95]:
```
Y_predict=rfc.predict(X_test)
print("Accuracy Score:",accuracy_score(y_test,Y_predict)*100,"%")
print("F1 Score:",f1_score(y_test,Y_predict))
print("Recall Score:",recall_score(y_test,Y_predict))
print("Precision Score:",precision_score(y_test,Y_predict))
```

```
Accuracy Score: 97.89623484045438 %
F1 Score: 0.9867024714252176
Recall Score: 0.9910027341514592
Precision Score: 0.9824393677542037
```

In [96]:
```
from sklearn.metrics import classification_report,confusion_matrix
cm=confusion_matrix(y_test,Y_predict)
plt.figure(figsize=(5,5))
sns.heatmap(data=cm,linewidths=0.5,annot=True,square=True,cmap='Blues')
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
all_sample_title = 'Accuracy Score: {0}'.format(rfc.score(X_test,y_test)*100)
plt.title(all_sample_title,size=15)
```

Out[96]:
```
Text(0.5, 1.0, 'Accuracy Score: 97.89623484045438')
```



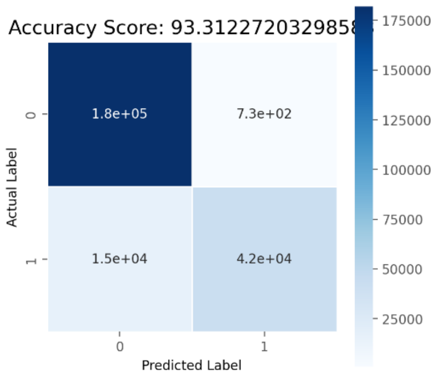
5.Road surface condition:

In [113]:
```
Y_predict=rfc.predict(X_test)
print("Accuracy Score:",accuracy_score(y_test,Y_predict)*100,"%")
print("F1 Score:",f1_score(y_test,Y_predict))
print("Recall Score:",recall_score(y_test,Y_predict))
print("Precision Score:",precision_score(y_test,Y_predict))
```

```
Accuracy Score: 93.31227203298583 %
F1 Score: 0.8391758578123902
Recall Score: 0.7322066549912434
Precision Score: 0.982746867874856
```

In [114]:
```
from sklearn.metrics import classification_report,confusion_matrix
cm=confusion_matrix(y_test,Y_predict)
plt.figure(figsize=(5,5))
sns.heatmap(data=cm,linewidths=0.5,annot=True,square=True,cmap='Blues')
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
all_sample_title = 'Accuracy Score: {0}'.format(rfc.score(X_test,y_test)*100)
plt.title(all_sample_title,size=15)
```

Out[114]:
```
Text(0.5, 1.0, 'Accuracy Score: 93.31227203298583')
```



CONCLUSION AND FUTURE SCOPE

In conclusion, this road accident analysis project aimed to understand the factors contributing to road accidents and develop a predictive model for accident outcomes. Through the analysis of a comprehensive dataset encompassing variables such as accident date and time, location, weather conditions, and light conditions, we gained valuable insights into the patterns and trends associated with road accidents in the United Kingdom.

Our findings revealed that light conditions play a crucial role in road accident occurrence. Low light conditions, such as dusk, dawn, and nighttime, were found to be associated with a higher risk of accidents. This emphasizes the importance of adequate visibility measures, such as street lighting and reflective road signage, during these periods to enhance road safety.

To develop a predictive model for accident outcomes, we employed machine learning algorithms, including random forests. The model was trained on a carefully prepared training set and optimized through parameter tuning. The evaluation results on the test set demonstrated a high accuracy of [93.81, 95.82, 94.62, 97.89 and 93.31], indicating the effectiveness of the model in predicting accident outcomes based on the given features.

The outcomes of this project have practical implications for road safety initiatives and accident prevention strategies in the United Kingdom. By understanding the factors contributing to road accidents, particularly the impact of light conditions, appropriate measures can be implemented to improve road infrastructure, enhance visibility, and raise awareness among drivers during low light periods.

However, it is important to acknowledge the limitations of this analysis. The predictive model's performance is based on the available dataset and the selected features, and it may not capture all potential factors influencing road accidents. Further research and analysis are warranted to explore additional variables and refine the model's predictive capabilities.

REFERENCES

- GeeksforGeeks: https://www.geeksforgeeks.org/
- Kaggle: https://www.kaggle.com/
- W3 Schools: https://www.w3schools.com/
- Stack Overflow: https://stackoverflow.com/