Comparing Traffic Accident Prediction to Codeless AI

CONTENTS

01 Problem Statement

Problem Statement

Big industry players providing cloud computing services such as Amazon and Google have been releasing codeless AI as a service lately.. The company is currently in the midst of expanding their infrastructure and the department is interested in looking whether it is worth to include such services .

With a provided sample data, they would like the following outcome to further expedite their decision:

1. Create a supervised classification model to predict accident severity and compare the outcome together with a codeless AI platform
2. Explore the related services provided and how it can benefit the team

02 About Google Console Platform

Google Cloud Platform

- **Google Cloud Platform** (**GCP**), offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products.
- They provides infrastructure as a service , platform as a service , and serverless computing environments.

Google Services used in this experiment

Data Studio

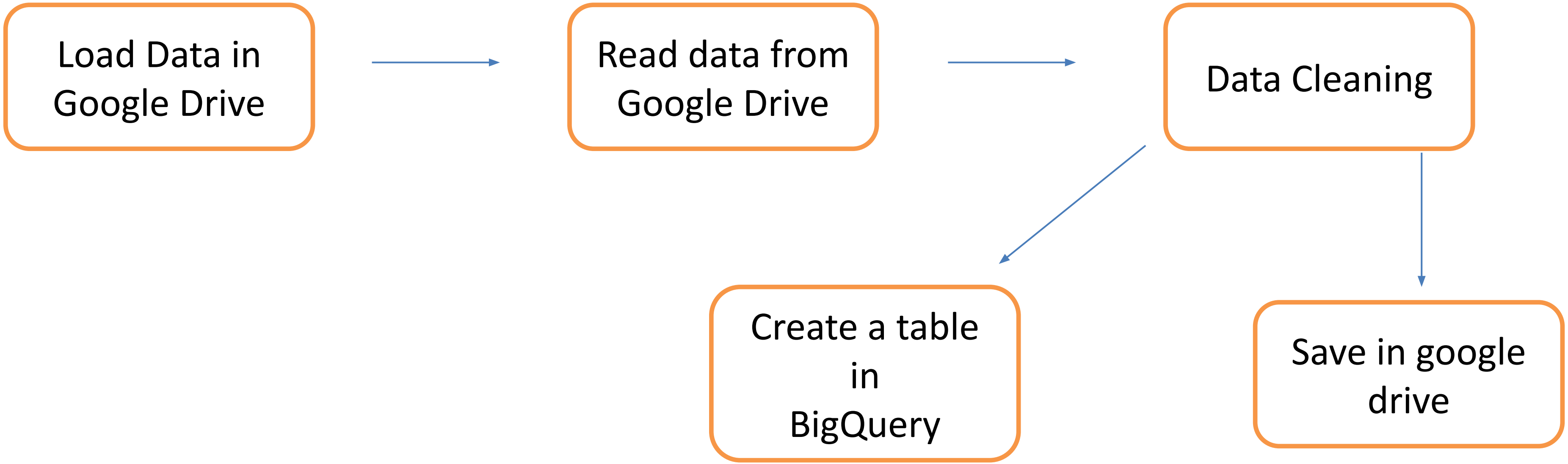
Vertex AI

BigQuery, Google Drive

Colaboratory

03 Data Findings and Preprocessing

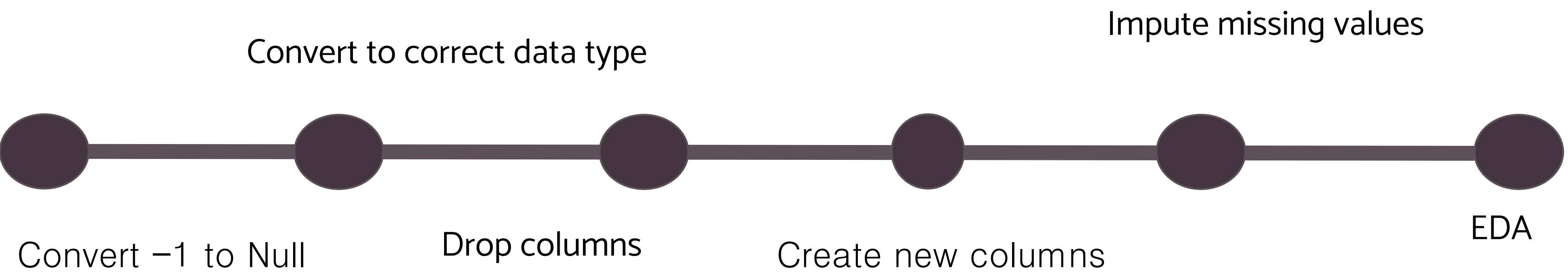
Data Loading Steps:



Data Findings

- Total of 91,199 records and 27 columns
- Missing data are indicated as -1
- There are columns that have values category "unknown"
  - Similar to rows where longitude and latitude that are null or it does not match to the category already indicated
  - Since it is in a small percentage, ignore.
- After data cleaning steps, have total of 23 columns

Data Cleaning



Drop columns

- Road_surface_conditions,special_conditions_at_site,carriageway_hazards second_road_class column data mostly belongs , pedestrian_crossing_human_control,pedestrian_crossing_physical_facilities, second_road_number data mostly belongs to “None”
- junction_control have more than 50% null values

Data Preprocessing

New columns created

1. Season
   - Based on day of year
2. Month
3. Hour
4. Co-ordinates
   - From latitude,longitude

Time

Missing Values

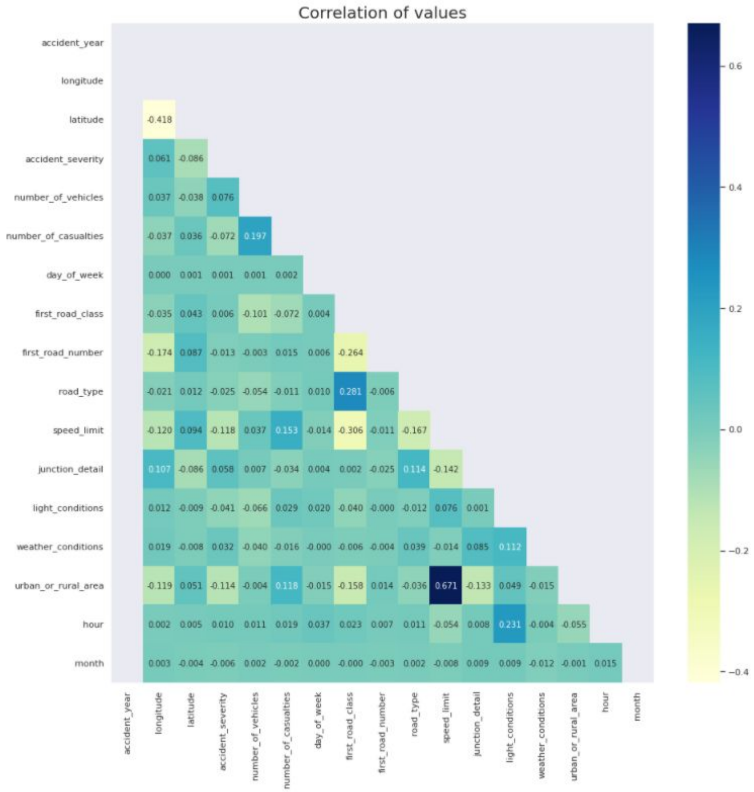
1. Speed limit
   - 10/12 of this null values belongs occurs in the urban area
   - Accidents found in urban area speed found to be mostly in speed limit of 30km/h, Rural areas at 60km/h
   - Impute accordingly based on the urban or rural areas
2. Light conditions
   - Use the date and hour to get light conditions
3. Junction detail
   - Uses median value to impute

4 . latitude and longitude

- Ignore as it is in very small percentage and won't be considered in model but to be used for data visualization

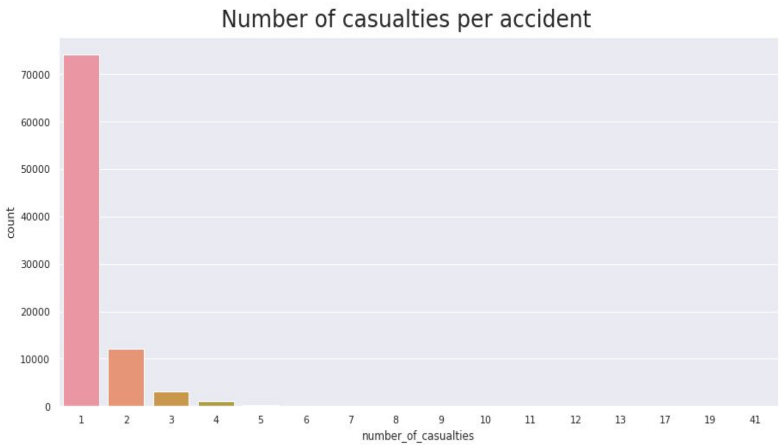
04 EDA

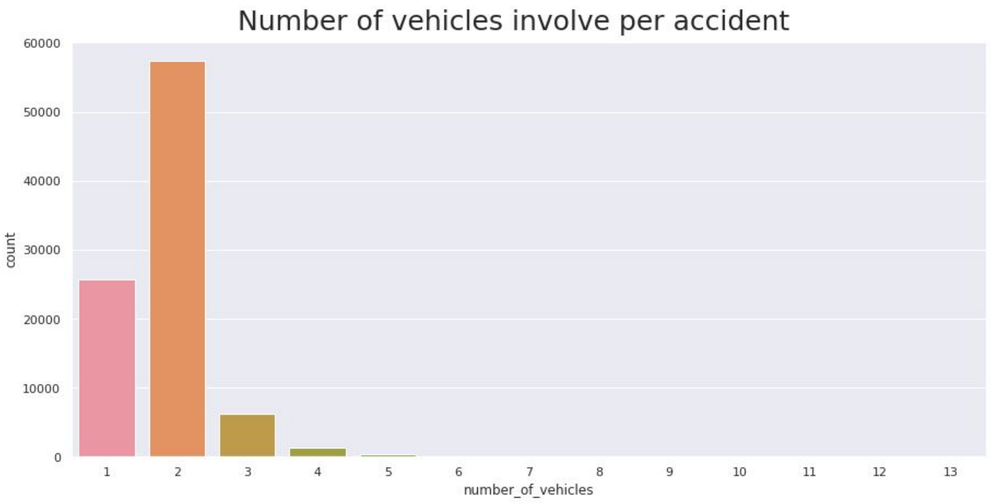
Correlation



- Values are not highly correlated except for urba_or_rural and speed
- This indicates that there is a higher speed limit in rural area

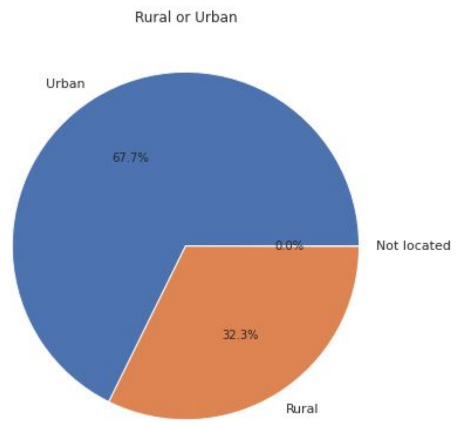
Casualties and vehicles involved

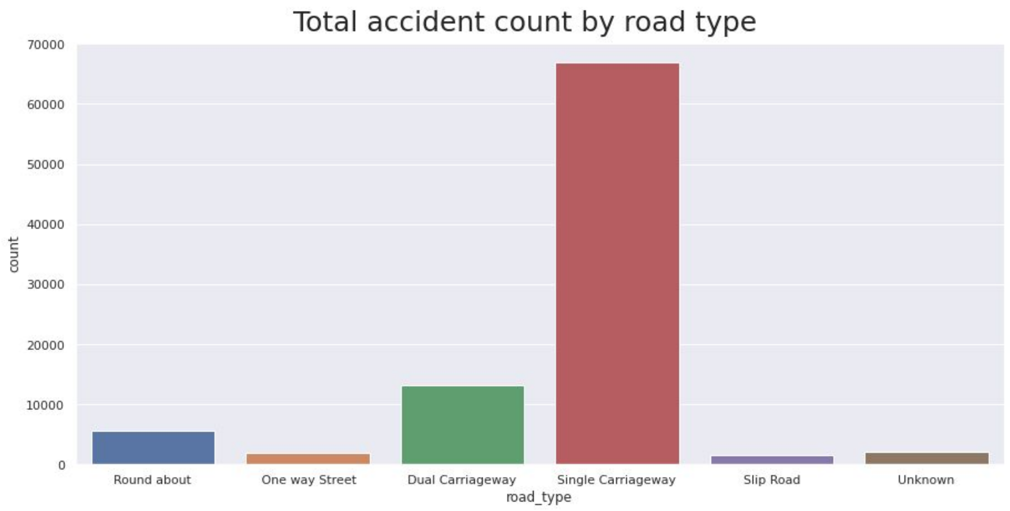




- Most accidents involves two vehicles with 1 casualty

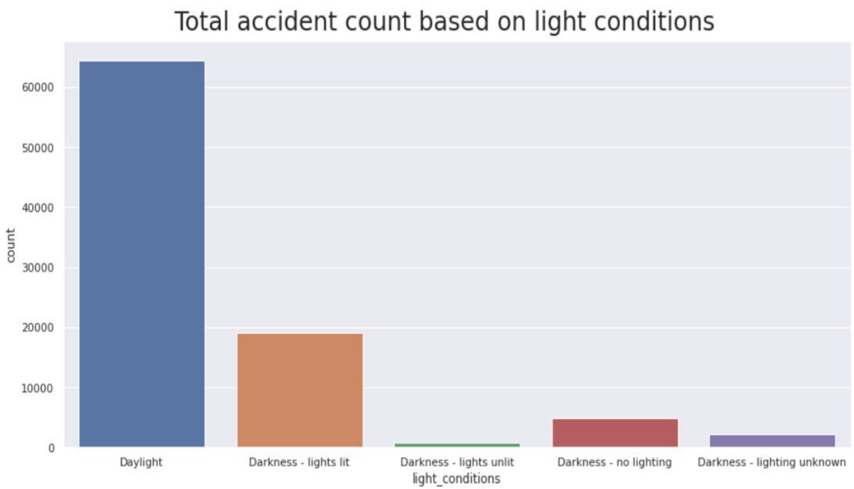
Road type and Urban or Rural

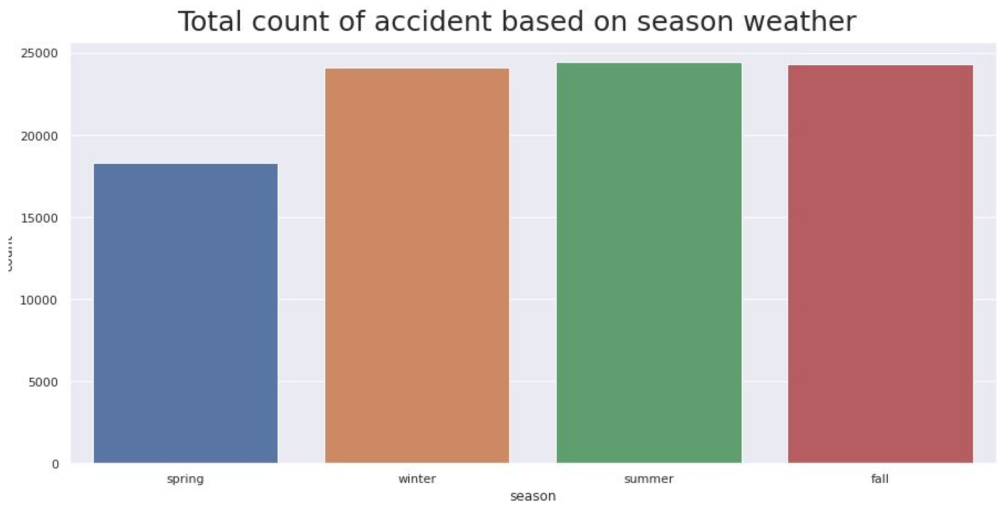




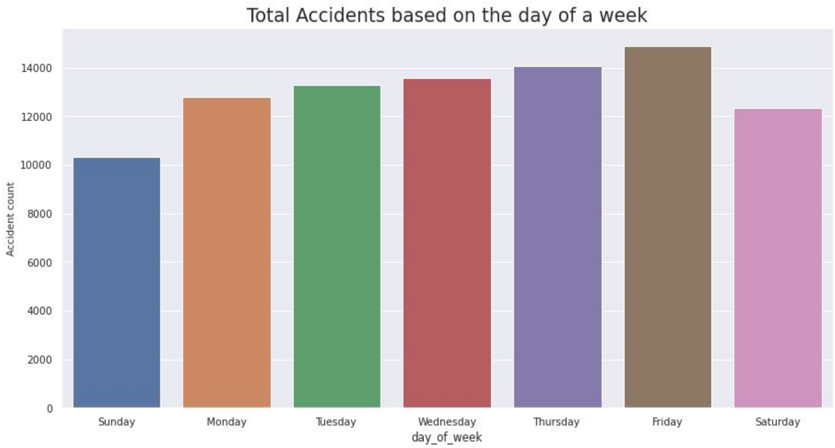
- Most accidents occur in urban areas and single carriageway

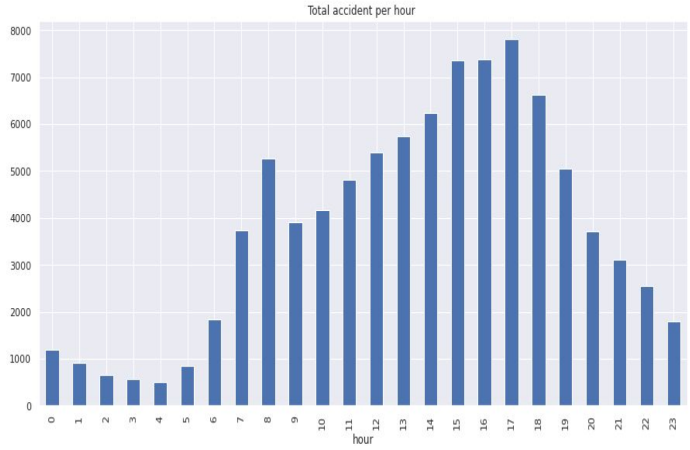
Light conditions and season





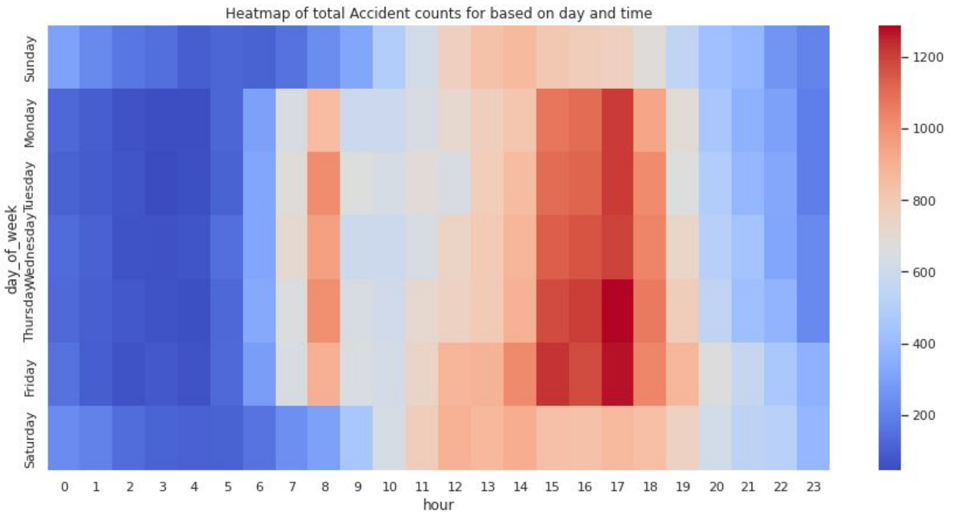
Time and Day

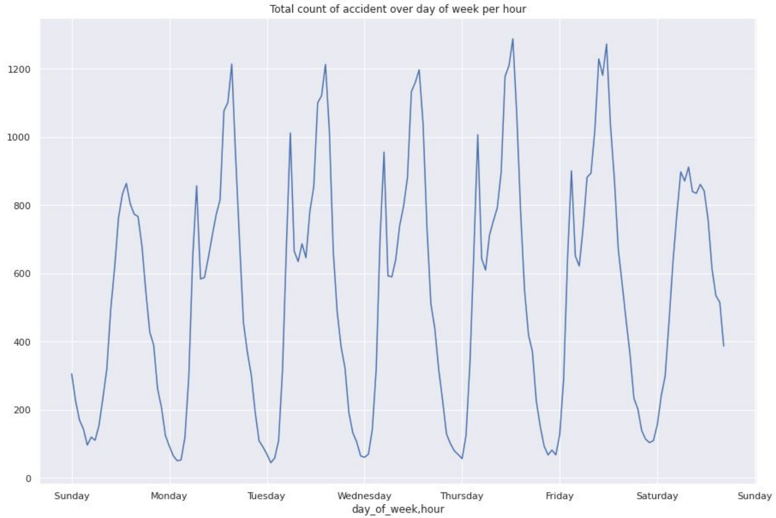




- Most accidents occur more on Fridays
- Evening peak hour tend to have more accidents than morning peak hour

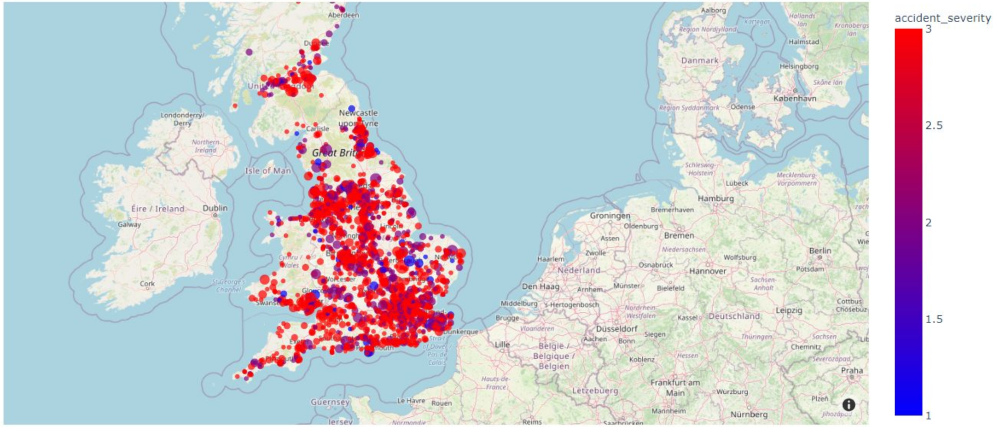
Time and Day





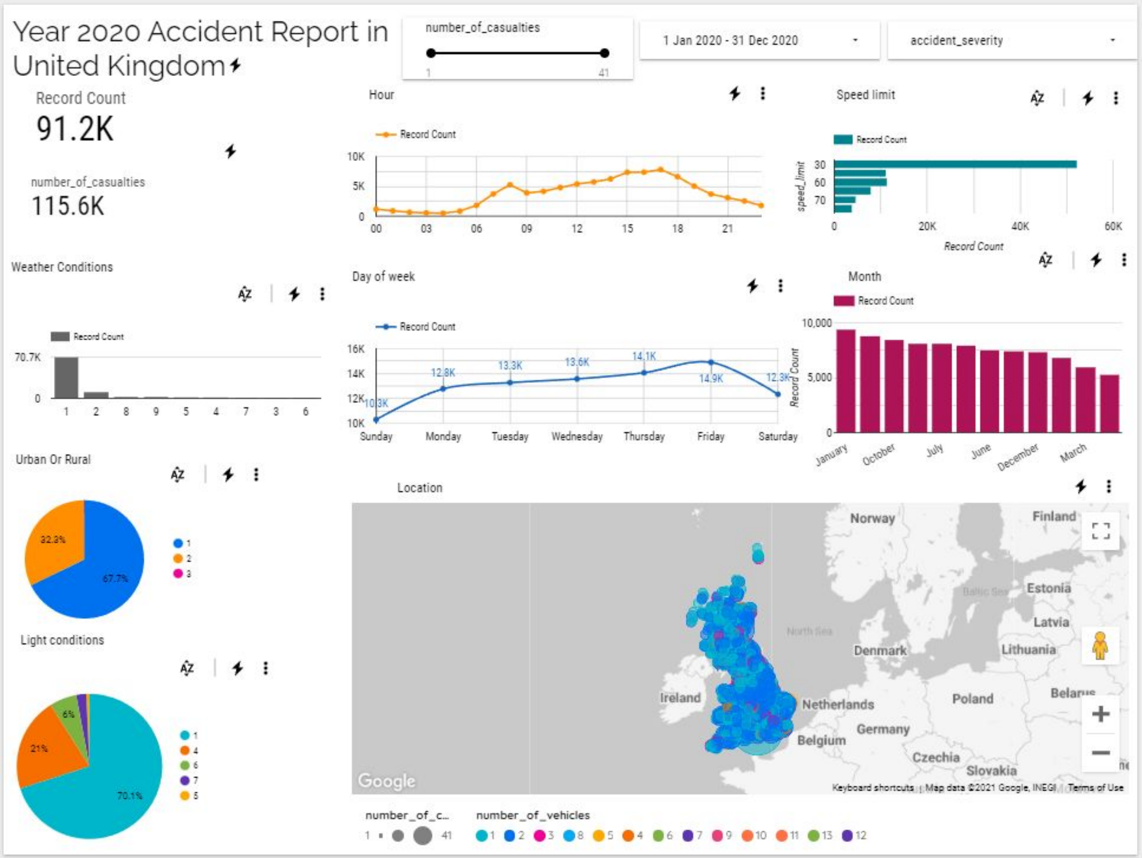
- Most accident occur between 4-6PM especially on
  thursdays and fridays

Location of accident during evening peak hour



- Heatmap shows accidents occur mostly in big cities like london,manchester,glasgow and birmingham
- Most accidents involve are however in "slight" category

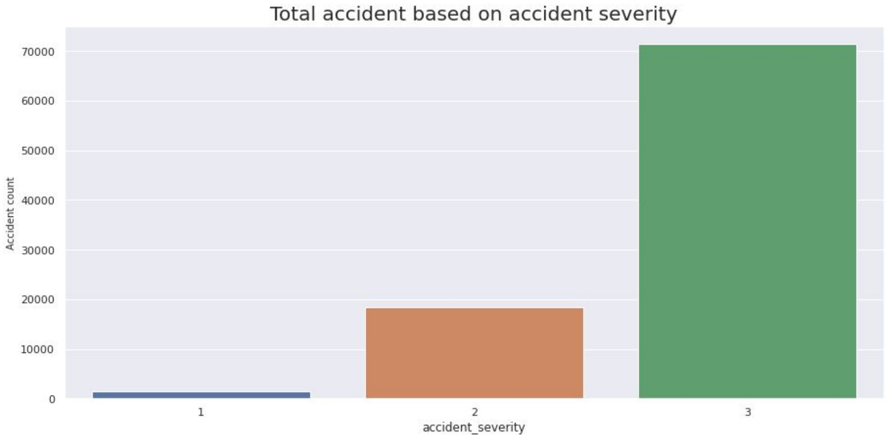
Dashboard



[Link]

04 Model

Checking of data imbalance



- Imbalance dataset for the target variable
- Upsampling is required to balance the dataset to fix imbalance dataset

Modelling

- Based on recent research in the same topic, multiple factors influence the accident severity
- Enable polynomial selection, feature selection, fix_imbalance
- Only uses 15 features out 23
- Uses Recall and F1 to select best model

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.7843	0.6321	0.3369	0.7022	0.6945	0.0133	0.0459	8.0833
gbc	Gradient Boosting Classifier	0.7810	0.6186	0.3395	0.6889	0.6951	0.0170	0.0415	123.0933
rf	Random Forest Classifier	0.7611	0.5889	0.3423	0.6739	0.6998	0.0322	0.0423	24.0000
et	Extra Trees Classifier	0.7360	0.5598	0.3430	0.6677	0.6937	0.0288	0.0321	25.6500
ada	Ada Boost Classifier	0.6999	0.5735	0.3574	0.6793	0.6888	0.0670	0.0674	9.7200
nb	Naive Bayes	0.6853	0.5360	0.3376	0.6164	0.6484	0.0007	0.0008	0.9933
dt	Decision Tree Classifier	0.6542	0.5238	0.3586	0.6708	0.6621	0.0412	0.0413	3.0033
knn	K Neighbors Classifier	0.5201	0.5422	0.3800	0.6782	0.5742	0.0416	0.0474	20.8500

Footnote: [link]()

Create and Tune Selected model

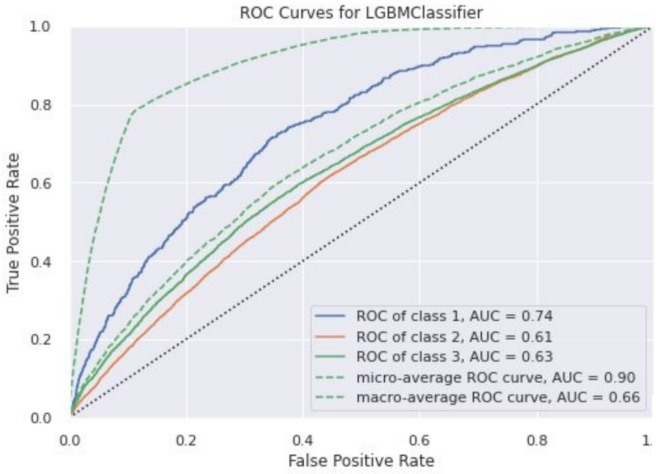
- From selected model, create the model and input the parameters to tune the model

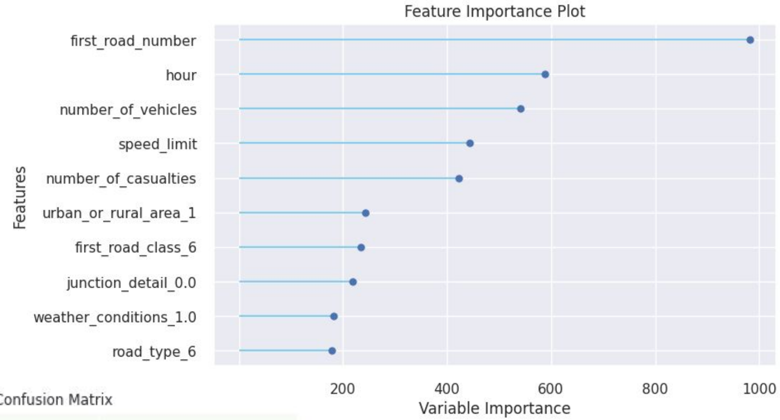
	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7856	0.6331	0.3400	0.7299	0.6960	0.0177	0.0642
1	0.7830	0.6257	0.3373	0.6869	0.6939	0.0113	0.0353
2	0.7846	0.6295	0.3390	0.7086	0.6962	0.0184	0.0568
Mean	0.7844	0.6294	0.3388	0.7085	0.6954	0.0158	0.0521

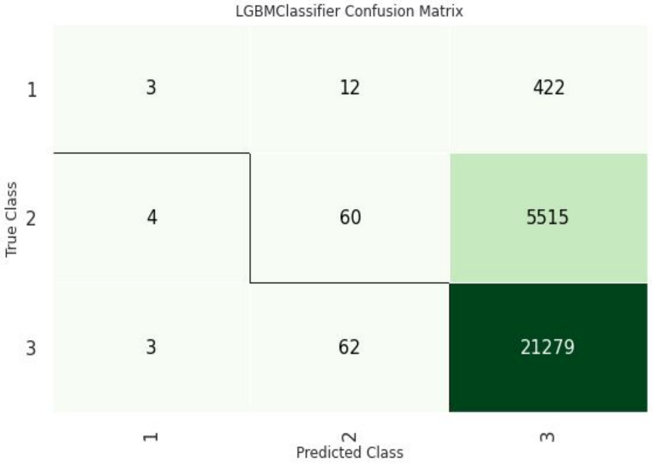
Test Data

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Light Gradient Boosting Machine	0.78	0.6307	0.3382	0.706	0.6882	0.0136	0.05

Metrics Output







Finalize and save model

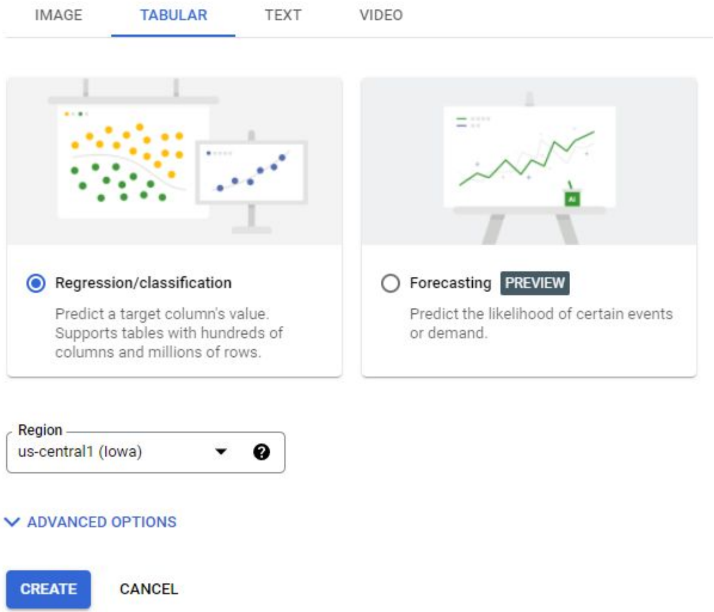
- Model is fit onto the complete dataset and include the test/hold-out sample to train the model

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Light Gradient Boosting Machine	0.7815	0.6588	0.3389	0.7455	0.6899	0.0186	0.0722

- Save the data that entire transformation pipeline for later use instead of going through the experiment all over again. This can be used to deploy in environments

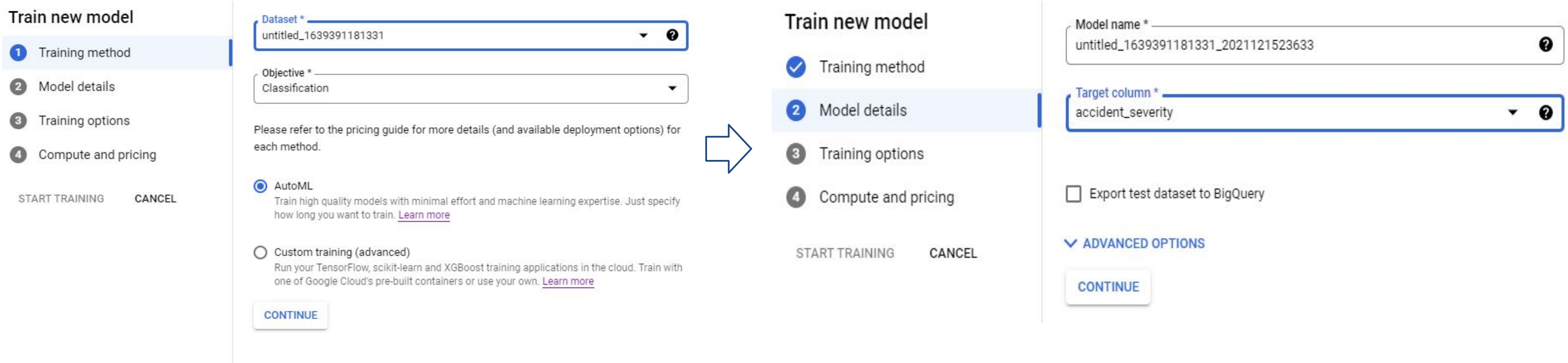
Vertex AI

- From the saved csv file in google drive, create a dataset by importing the csv file from google drive and choose the model type



[Link]

Vertex AI



- Select AutoML to use Google's codeless AI platform
- Custom training can be selected only for existing deployed python applications in containers

Vertex AI

Filter Enter property name or value

	Column name	Transformation	Missing % (count)	Distinct values	Correlation w/target	
☐	accident_index	Numeric	-	91199	-	⊕
☐	accident_reference	Numeric	-	91199	-	⊕
☐	accident_severity Target		-	3	-	
☐	accident_year	Categorical	-	1	-	⊕
☐	date	Timestamp	-	366	-	⊕
☐	day_of_week	Categorical	-	7	-	⊖
☐	first_road_class	Categorical	-	6	-	⊖
☐	first_road_number	Numeric	-	3068	-	⊖
☐	hour	Numeric	-	24	-	⊖
☐	junction_detail	Categorical	-	10	-	⊖
☐	latitude	Numeric	-	88749	-	⊕
☐	light_conditions	Categorical	-	5	-	⊖
☐	longitude	Numeric	-	89590	-	⊕
☐	month	Numeric	-	12	-	⊖
☐	number_of_casualties	Numeric	-	16	-	⊖
☐	number_of_vehicles	Numeric	-	13	-	⊖
☐	road_type	Categorical	-	6	-	⊖
☐	season	Categorical	-	4	-	⊖
☐	speed_limit	Categorical	-	6	-	⊖
☐	time	Timestamp	-	1438	-	⊕
☐	urban_or_rural_area	Categorical	-	3	-	⊖
☐	weather_conditions	Categorical	-	9	-	⊖

Total of 15 feature columns are included in the training

ADVANCED OPTIONS

CONTINUE

Total of 21 feature columns are included in the training

Weight column

Select a column

Optimization objective

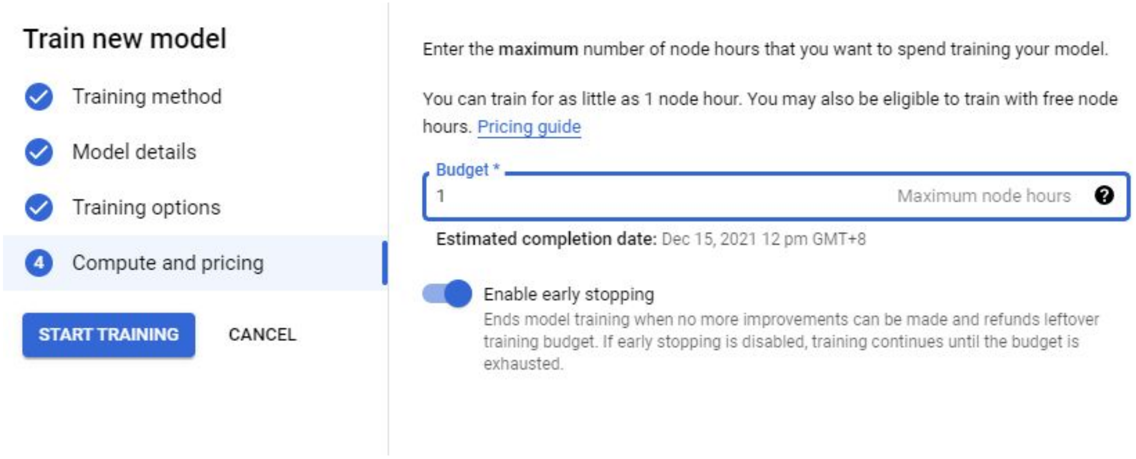
- ○ AUC ROC
  Distinguish between classes
- ◉ Log loss
  Keeps prediction probabilities as accurate as possible
- ○ AUC PRC
  Maximize precision-recall for the less common class
- ○ Precision — At recall
- ○ Recall — At precision

SHOW LESS

CONTINUE

Select features for the model

Vertex AI



Link

Vertex AI

Filter Filter labels

Confidence threshold 0.5

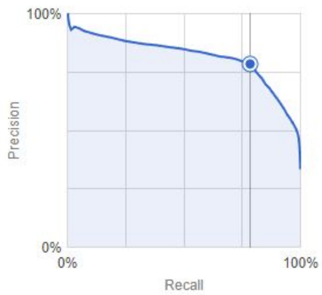
All labels	0
3	0.86421
2	0.28407
1	0.06593

All labels

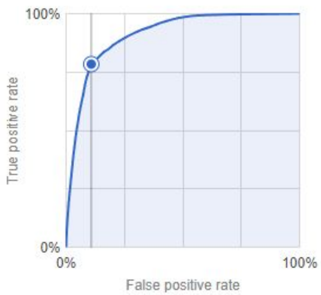
PR AUC	0.815
ROC AUC	0.91
Log loss	0.549
F1 score	0.7842537
Precision	78.5%
Recall	78.4%
Created	14 Dec 2021, 01:08:10

To evaluate your model, set the **confidence threshold** to see how precision and recall are affected. The best confidence threshold depends on your use case. Read some example scenarios to learn how evaluation metrics can be used.

Precision-recall curve



ROC curve



Precision-recall by threshold

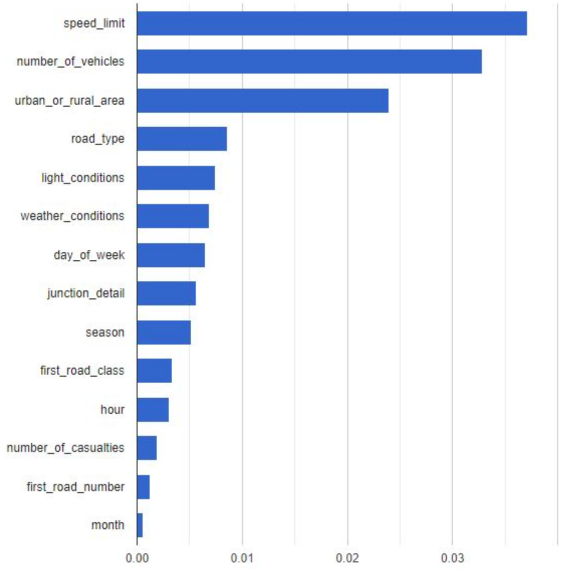
Vertex AI

Confusion matrix

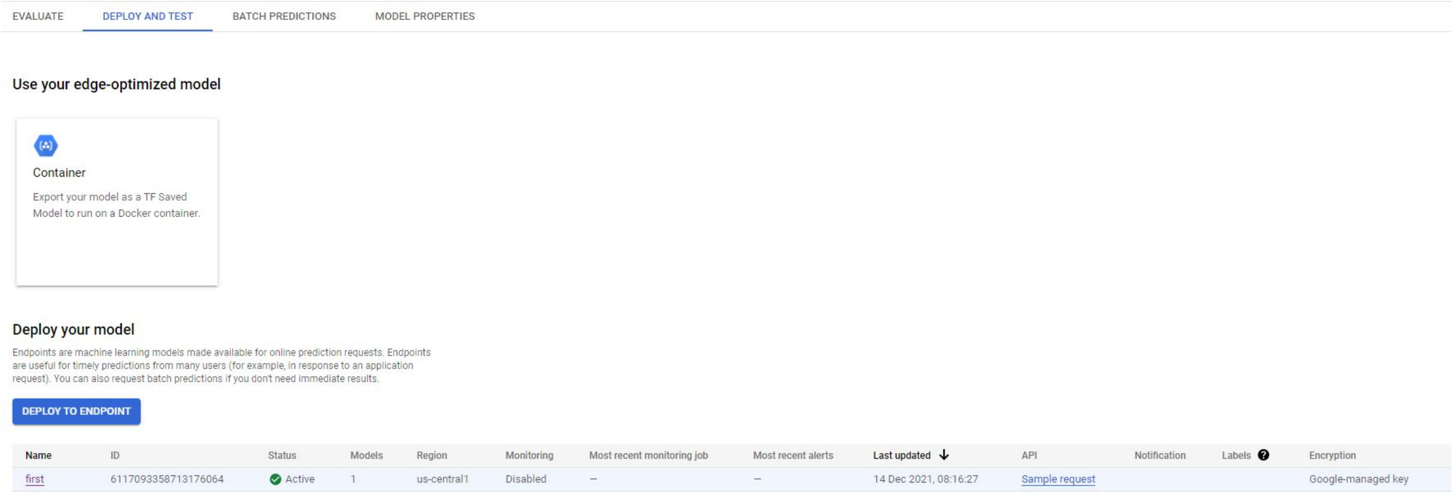
This table shows how often the model classified each label correctly (in blue), and which labels were most often confused for that label (in grey).

True label \ Predicted label	3	2	1
3	7172	—	—
2	1835	0	—
1	135	—	0

Feature Importance



Deploy and test model



- Can choose to deploy and test predictions online for many users

05 Comparison

Comparison of results

Model	AUC	Recall	Prec.	F1
Light Gradient Boosting Machine(Train)	0.6294	0.3388	0.7085	0.6954
Light Gradient Boosting Machine(Test)	0.6588	0.3389	0.7455	0.6899
Google Vertex AI AutoML	0.784	0.3844	0.785	0.784

- Loss between the train and test chosen model is very little for the created model
- Google Vertex AI AutoML has higher precision and f1 score comparing to the chosen model.
- However, there are disadvantages in using such codeless model

Findings of using Vertex AI AutoML

1. Easy to create, just clicks away
2. Lack of customization and control on what goes into the model
3. Time taken to run the model can be longer than expected
4. No transparency what was done by the model
5. Unable to use hyperparameter tuning to further enhance
6. Unable to enable properties such as feature selection,polynomial features or SMOTE
7. Limits to data size at 1M
8. For classification prediction, only minimize-log-loss is support to optimize which may not always support a project's objective
9. For imbalance class, best practice stated was to have at least 100 rows of data for every class and assign a manual split to make sure enough rows with the minority outcomes are included in every split.

05

Conclusion

Conclusion

- Cloud computing platforms such as Google cloud platform provide many services that allow users retrieve data online and collaborate to perform business activities
- Cloud services does benefit the analytics team as many related services can be found in one platform
- However, there are many limitations using such codeless AI services for machine learning despite the ease of creating models by just few clicks
- Try other models to compare the efficient of the AutoML

Suggestion

- Continue using python libraries available to create predicting models that allows transparency, control and enhancement to the model that may better fit the objective of project
- Deploy them in a cloud computing service so that similar experiment does not have to be re-created and many users can use the models at one time to predict or collaborate on experiment
    - Also allows to create custom models which enables certain properties to further tune model

THANK YOU!