Big Data and Data Mining.

Workshop Number 4.

(17 & 18/March/2022).

Aims of the workshop.

This workshop will firstly consolidate some of the learning from the previous week before starting to build a model with the Traffic dataset.

Workshop Timetable.

The timetable below should be taken as indicative only. We will modify the times according to how quickly things progress.

|  |  |
| --- | --- |
| **Time** | **Activity** |
| 13:00-13:05 | Welcome. |
| 13:05-13:30 | Reflection on learning: Outliers. |
| 13:30-14:30 | Afternoon Exercises 1. |
| 14:30-14:45 | Discussion / Tutorial on Afternoon Exercises. |
| 14:45-15:45 | Afternoon Exercises 2. |
| 15:45-16:00 | Wrap-up Discussion. |

Useful Information.  
Throughout this workshop you may find the following useful.

Python Documentation

https://docs.python.org/3.8/

This allows you to lookup core language features of Python 3.8 as well as tangential information about the Python Language. We will refer you back to the Programming module for more notes on this.

Jupyter Notebook Basics

Jupyter itself offers some basic documentation for people new to the editor. These can be found on https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics. html

Jupyter can also use markdown cells for text input to describe things. If you wish to annotate each cell as to which Exercise it belongs to, you may find https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20 Markdown%20Cells.html​ useful.

Reminder.

We encourage you to discuss the contents of the workshop with the delivery team, and any findings you gather from the session.

Workshops are not isolated, if you have questions from previous weeks, or lecture content, please talk to us.

The contents of this workshop are ​not​ intended to be 100% complete within the session; as such it’s expected that some of this work be completed outside of the session. Exercises herein represent an example of what to do; feel free to expand upon this.

**13:00. Welcome.**

Dr. Kevin Pimbblet. ([k.pimbblet@hull.ac.uk](mailto:k.pimbblet@hull.ac.uk))

**13:05. Discussion (mini-tutorial / study group) on Week 7’s learning.**

In Week 7, we have begun to look at Outliers as our main topic – this will be continued in Week 8. For now, I’d like to review our general thoughts about outliers.

Prompter Questions for Discussion:   
(a) What is an outlier anyway?

(b) What should we do with outliers? Should they always be retained? Why or why not?

Changing the topic to the PROJECT now. For discussion is the next point:  
(c) Which variables (or features) have the most discriminative power to predict fatalities in your opinion? Any thoughts on how this could be investigated?

**13:20. Afternoon Exercises, Part 1.**We will start our afternoon exercises by looking at which variables actually are the best at predicting fatalities within the accident database.

Exercise 1. Coding.

***SelectKBest METHOD! This will be returned to in Week 9 lectures.***

Read in the accident database using Pandas. Then type:

**from sklearn.feature\_selection import SelectKBest, chi2, f\_regression, f\_classif**

Drop any columns that you have not cleaned up or that still have NaNs within. You can return to imputation at a later date if needed.

We will need to specify some targets:

**predictors = ["Weather\_Conditions", "Speed\_limit", "Road\_Surface\_Conditions", "Light\_Conditions"]**

We will also need to get rid of negative values as these can cause problems.

**usethis = df.dropna()**

**usethis.reset\_index(drop=True)**

**usethis = usethis.filter(['Weather\_Conditions','Speed\_limit','Road\_Surface\_Conditions', 'Light\_Conditions', 'Accident\_Severity'], axis=1)**

**usethisnow = usethis[(usethis['Weather\_Conditions']>0) & (usethis['Speed\_limit']>0)**

**& (usethis['Road\_Surface\_Conditions']>0) & (usethis['Light\_Conditions']>0)]**

**usethisnow**

Now let’s see what the top N conditions are for our target variable (i.e., accident severity).

**# Perform the feature selection**

**selector = SelectKBest(f\_classif, k="all")**

**selector.fit(usethisnow[predictors], usethisnow["Accident\_Severity"])**

**# Here we will get the p-values for each feature and then transform from p-values into some scores**

**scores = -np.log10(selector.pvalues\_)**

**# Use plt to plot the scores!**

**plt.bar(range(len(predictors)), scores)**

**plt.xticks(range(len(predictors)), predictors, rotation='vertical')**

**plt.show()**

You should have obtained a graph like this one.

Chart

Description automatically generated

This shows that Speed\_limit is the main factor out of the ones that we have looked at.

Question: are you surprised by this at all?

Exercise 2.

Expand the features that you are considering. Which ones at the top X (where X might be 4, 5, or so)?

Exercise 3.

Apply this approach to the other files – i.e., the vehicle information file.

Put in chat any graphs you produce and comment which features are the “best” for predicting accidents.

**14:30. Afternoon Discussion.**  
In the first half of today’s exercises, we looked at identifying which features are the most important. We could, of course, use feature engineering to combine some features together if we wanted to.

Before we begin this afternoon, I want to have a discussion on which features you think are the most important, and whether you are surprised at all (or not).

**14:45. Afternoon Exercises.**

In the afternoon exercises, we will create a model to predict which class certain accidents belong to and quantify our confidence in them.

Exercise A.

**Choose the features that are important.**

You will have your own list of which features you think are important for predicting the severity of accidents. In the exercises that follow, I will be using:

**Weather\_Conditions, Speed\_limit, and Light\_Conditions.**

As you go through the exercises that follow, feel free to replace these three with others of your choosing!

Exercise B.

We are now going to build our model. To do this, I’m choosing to use a Decision Tree. We can replace this with other methods, of course, to experiment!

First we need to do the usual data hygiene approach of splitting our data into test and train.

**from sklearn.model\_selection import train\_test\_split**

**y\_target = usethisnow["Accident\_Severity"].values**

**Xfeatures = usethisnow.drop("Accident\_Severity", axis=1)**

**X\_train, X\_validation, y\_train, y\_validation = train\_test\_split(Xfeatures,y\_target,test\_size=.25,random\_state=1)**

Exercise C.

Let’s build the model now.

**from sklearn import tree**

**from sklearn.tree import DecisionTreeClassifier**

**decision\_tree = DecisionTreeClassifier(criterion="gini")**

**decision\_tree.fit(X\_train, y\_train)**

**Y\_pred = decision\_tree.predict(X\_validation)**

This next line computes the accuracy of the model.

**acc\_decision\_tree = round(decision\_tree.score(X\_train, y\_train) \* 100, 2)**

I scored 79%. What accuracy did you obtain with your features? Let us know in chat.

Exercise D (time permitting).

We can also get the probabilities of our models.

**decision\_tree.predict\_proba(X\_validation)**

The thing is, my decision tree predicted almost everything as a severity level 3 incident. It got most right as well. Without really trying.

This tells me that I need to modify my model since all its doing is latching on to speed\_limit.

I need different models or features to better decide these things.

I encourage you to do this by replacing the decision tree with naïve Bayes (or other models), and to play around with which features you will use.

In the next workshop we will look at multiple models and combining them together. But for now: we’ve made our first full model and are now in an excellent position to complete the Project. **i.e., at this stage you could already start writing it all up.**

But more next time!