[A2] DataScience& MachineLearning

Dealing with Imbalanced Dataset

Practical Assignment

余知諺 104403039

Google colab 連結

Boys and Girls

- https://colab.research.google.com/drive/1jJDBz3EczhEa_h3zx2bc4UqYDVcpZpEb#scrollTo=FKs_-3GedLQ6

Bank marketing

- https://colab.research.google.com/drive/1pQS7Jz2Tp-c9AXQBPaTk61tE_m-YJ9EM

Experiment overview

What we are going to do ?

在google colab上實作處理imbalanced dataset

Dataset(皆為binary class)

1.Boys and girls datasets

2.Bank Marketing Data Set

Brief intro: 兩類別的數量比約為13:87，此數據及與葡萄牙銀行機構的營銷活動有關蒐集的數據包含age、job、education、house、loan、default、marital等銀行客戶相關資訊和多項行銷相關、社會背景相關或是其他屬性，並透過這些屬性來判斷客戶**是否會 訂購定期存款**。

Experiment overview

HOW?

Classifiers➔選用在assignment 1 我們在資料中所查到適合不平衡數據集且效果好的algorithm

- Decision Tree
- Random forest

➔實作驗證這兩種適合的分類器是否有符合資料上的內容和效果

Methods for dealing with imbalanced dataset:

- Random undersampling
- Random oversampling

➔透過Random resample(增加minority/減少majority)來讓不平衡數據集的資料平衡以便機器學習

Flow of experiments

step1.導入Dataset 並進行讀取

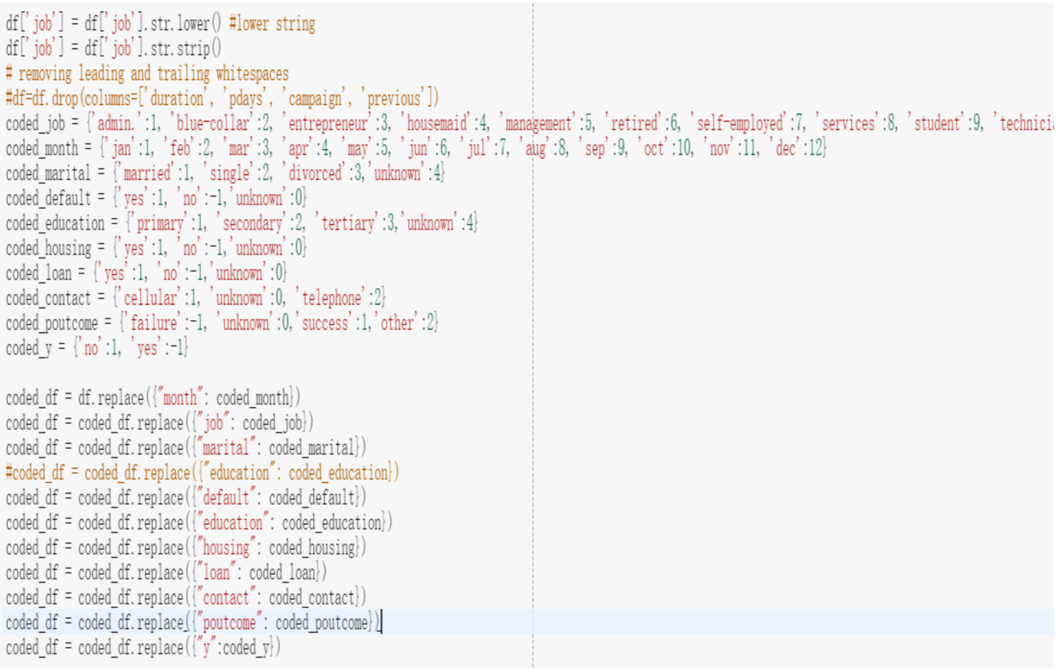
step2. 對Dataset進行 preprocess

- Boys and girls datasets

➔做字型調整、Drop columns(id, timestamp, self intro) transform、Outlier removals、Train/test set split

- Bank Marketing Dataset

➔Transform、 Train/test set split



```
df['job'] = df['job'].str.lower() #lower string
df['job'] = df['job'].str.strip()
# removing leading and trailing whitespaces
#df=df.drop(columns=['duration', 'pdays', 'campaign', 'previous'])
coded_job = {'admin.':1, 'blue-collar':2, 'entrepreneur':3, 'housemaid':4, 'management':5, 'retired':6, 'self-employed':7, 'services':8, 'student':9, 'technici
coded_month = {'jan':1, 'feb':2, 'mar':3, 'apr':4, 'may':5, 'jun':6, 'jul':7, 'aug':8, 'sep':9, 'oct':10, 'nov':11, 'dec':12}
coded_marital = {'married':1, 'single':2, 'divorced':3,'unknown':4}
coded_default = {'yes':1, 'no':-1,'unknown':0}
coded_education = {'primary':1, 'secondary':2, 'tertiary':3,'unknown':4}
coded_housing = {'yes':1, 'no':-1,'unknown':0}
coded_loan = {'yes':1, 'no':-1,'unknown':0}
coded_contact = {'cellular':1, 'unknown':0, 'telephone':2}
coded_poutcome = {'failure':-1, 'unknown':0,'success':1,'other':2}
coded_y = {'no':1, 'yes':-1}

coded_df = df.replace({"month": coded_month})
coded_df = coded_df.replace({"job": coded_job})
coded_df = coded_df.replace({"marital": coded_marital})
#coded_df = coded_df.replace({"education": coded_education})
coded_df = coded_df.replace({"default": coded_default})
coded_df = coded_df.replace({"education": coded_education})
coded_df = coded_df.replace({"housing": coded_housing})
coded_df = coded_df.replace({"loan": coded_loan})
coded_df = coded_df.replace({"contact": coded_contact})
coded_df = coded_df.replace({"poutcome": coded_poutcome})
coded_df = coded_df.replace({"y":coded_y})
```

***Preprocess用意:**

- 避免數據集中極端值或誤輸入的資料影響train的情況
- 字元轉換成數值才能使訓練器可以學習

Flow of experiments

Step3

對Dataset 切出train/test set

Step4

對要train的資料進行 random under sampling/oversampling 的重採樣

➔使不平衡比例的兩類數據能夠在相同數量的情況下進行訓練

而不會使訓練器只學習到　偏向比例較多的那一方

Step5

建立decision tree 和 random forest的classifier

Step6

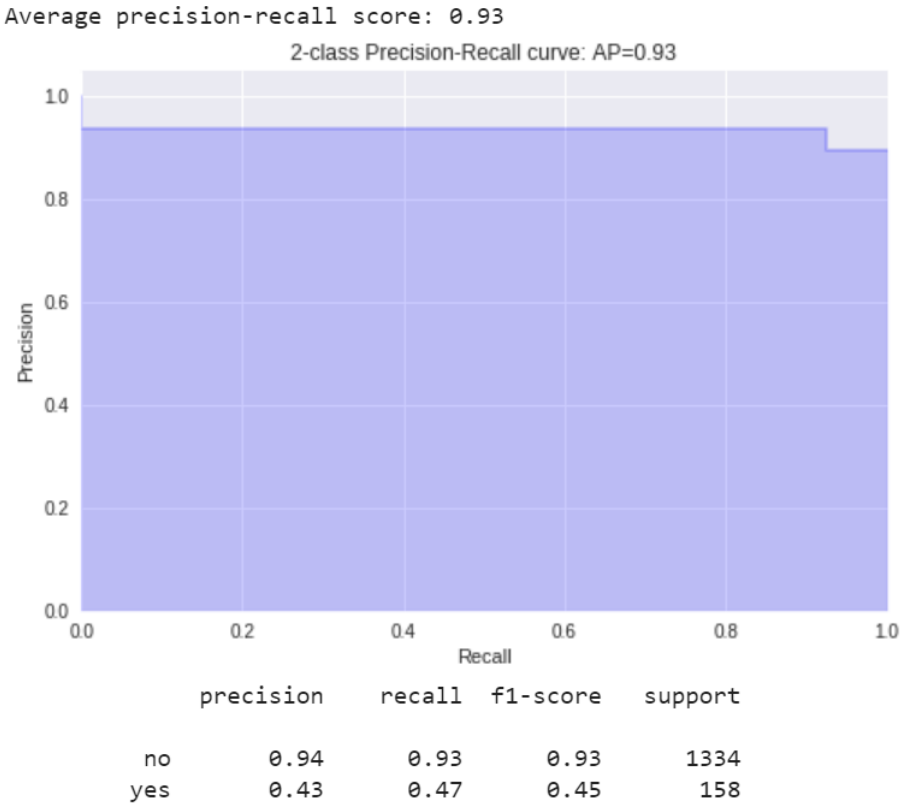
將original 與 resample 後的資料分別倒入兩classifier中，圖像化其結果

Original datasets vs “Processed” datasets

以Bank marketing 的 dataset 為例

Original dataset

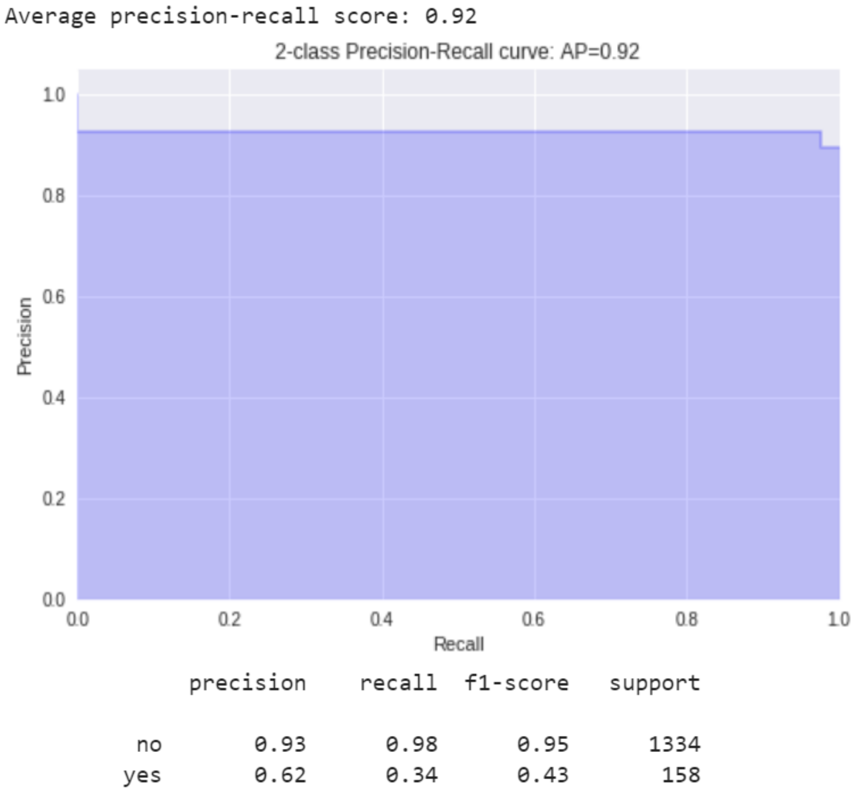
Decision tree



```
Average precision-recall score: 0.93
```

```
             precision    recall  f1-score   support

         no       0.94      0.93      0.93      1334
        yes       0.43      0.47      0.45       158
```

Random forest



```
Average precision-recall score: 0.92
```

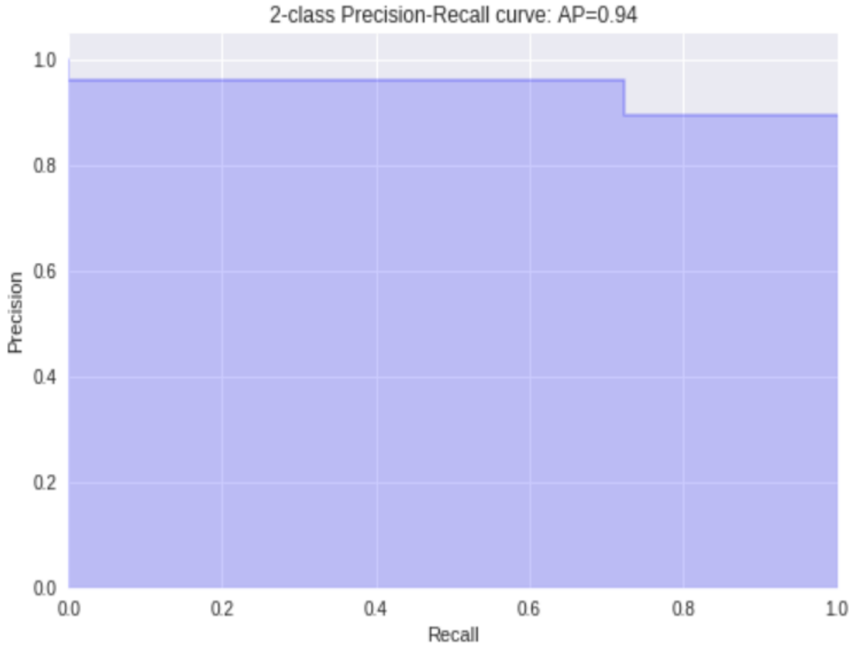
```
             precision    recall  f1-score   support

         no       0.93      0.98      0.95      1334
        yes       0.62      0.34      0.43       158
```

Original datasets vs “Processed” datasets

Processed dataset(under sampling)

Decision tree

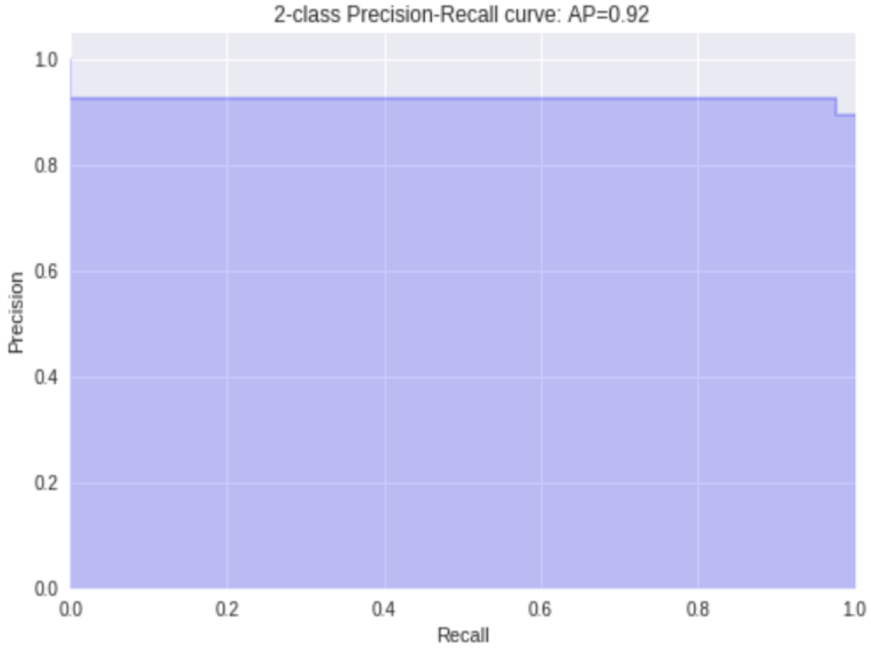
```
Average precision-recall2 score: 0.94
```



```
             precision    recall  f1-score   support

         no       0.96      0.72      0.83      1334
        yes       0.25      0.77      0.37       158
```

Random forest

```
Average precision-recall score: 0.92
```

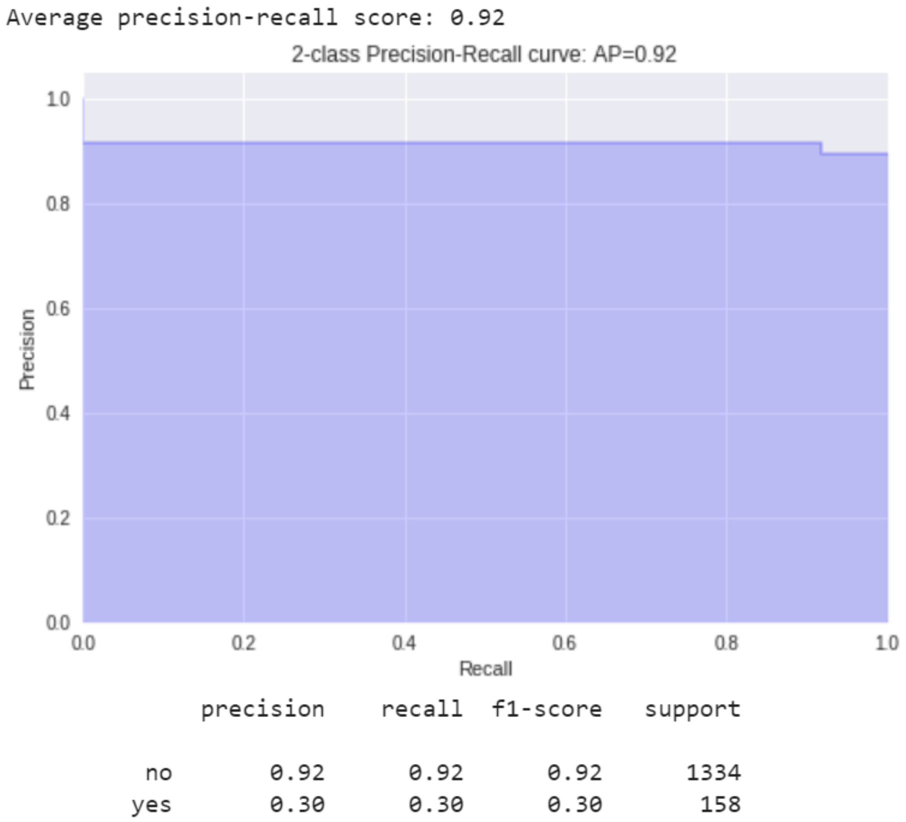


```
             precision    recall  f1-score   support

         no       0.98      0.80      0.88      1334
        yes       0.34      0.86      0.49       158
```

Original datasets vs “Processed” datasets

Processed dataset(oversampling)

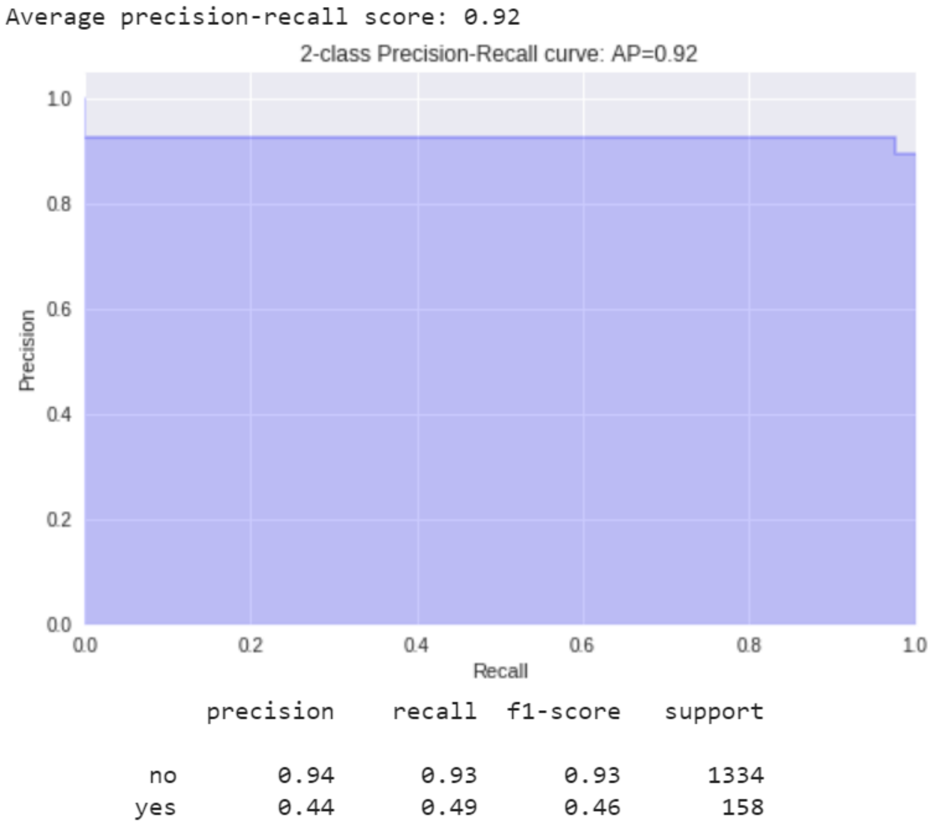
Decision tree



```
Average precision-recall score: 0.92
```

```
             precision    recall  f1-score   support

         no       0.92      0.92      0.92      1334
        yes       0.30      0.30      0.30       158
```

Random forest



```
Average precision-recall score: 0.92
```

```
             precision    recall  f1-score   support

         no       0.94      0.93      0.93      1334
        yes       0.44      0.49      0.46       158
```

Original datasets vs “Processed” datasets

Summary:

1. processed過的dataset 在decision tree 的表現

2.在很多資料文獻中都說明decision tree 和 random forest 的效果對處理imbalanced data 的效果都很好，而其中又以random forest 的效果比decision tree更好 但在實際套入decision tree 和 random forest 的結果下發現雖說兩者的都分數都很高，但random forest跑出來的分數並不比decision tree的分數高，甚至在兩個實作dataset中，decision tree的表現更好

Research Questions

Q1.What classifiers work the best with imbalanced datasets “off the shelf”?

A: 在未修改的情況下，Random Forest work the best

Q2. What methods best “offset” the effect of imbalanced datasets?

A:我會選擇oversampling ，雖然under sampling 跑出來的分數也高，但是要考慮到info loss 的問題，相比oversampling 的 overfitting的問題更加嚴重，故我認為oversampling best “offset” the effect of imbalanced datasets

Research Questions

Q3. Binary class vs Multi-class problems?

A:

- Binary:

情況較簡單，在只需區分兩種結果的情況下，而在解決了二分類中的數據不平衡問題後，推廣就能得到多分類(multi-class)情況下的解決方案。

- Multi-class:

多分類遇到的問題較為複雜，在考慮不平衡的數據比例，就會發現不只一類需要做調整，導致在調整數據集和建立分類器的複雜度和難度就會提升

Research Questions

Q4. Classification vs Regression?

A:

. Classification

將輸入變量（X）到**離散**輸出變量（y）的映射函數（f）的任務，輸出變量通常稱為標籤或類別，用來預測離散類標籤。

➔像我們做的decision tree 和random forest 就是對label 做分類來找判斷值

. Regression

將輸入變量（X）的映射函數（f）**連續**輸出變量（y）的任務。用來預測連續數量。通常以假設它們兩者之間有線性關係，但因為在做imbalanced dataset的處理時用regression的來訓練時會導致訓練器會偏向較大比例差距的分類，故效果不好

➔處理imbalanced dataset時還是以classification的方法較佳

Research Questions

Q5. Any questions that you want to investigate.

A:

1.

原先在做resampling時有考慮到底要先做resampling再切還是先切再做resampling，後來有發現到如果先resample 再切時會去影響到test的數量，訓練出來的值就不客觀了，而先切再對train set做resample 就可以達到我要的效果，能以平衡的數量去做訓練之外，也不會影響到test set的值

2.

在與其他組員討論時，我們也有試著作出SVM的模型作來train看看imbalanced data，結果發現one class SVM的效果和分數遠低於Decision tree 和 Random forest，這與我們當初所作文獻調查發現的one class SVM也適合做imbalanced data 的論點有所出入，不知是one-class SVM事實上不適合做Imbalanced data 還是我們的作法和調整參數的地方有所錯誤?

Reference

- https://www.zhihu.com/question/269698662(undersampling和oversampling會對模型帶來怎樣的影響)
- https://imbalanced-learn.readthedocs.io/en/stable/under_sampling.html (under-sampling技術實作)
- https://bigdatafinance.tw/index.php/tech/data-processing/353-2017-03-28-11-36-54(二分類與多分類處理關係)
- https://data.world/data-society/bank-marketing-data (Bank Marketing 預測)
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (Random forest實作)
- https://www.kaggle.com/residentmario/undersampling-and-oversampling-imbalanced-data(undersampling and oversampling data 實作)
- https://www.kaggle.com/rishabh8492/once-class-svm-to-detect-anomalySVM (SVM 實作)
- https://stackoverflow.com/questions/50999596/smote-oversampling-on-text-classification-in-python (SMOTE Oversampling作法)
- https://www.cnblogs.com/pinard/p/6160412.html (random forest skit-learn)
- https://towardsdatascience.com/a-guide-to-decision-trees-for-machine-learning-and-data-science-fe2607241956?fbclid=IwAR3yUvdGL5RQ9YZx1zX0vTD4wbxHZTohO_SccPLQeL_XOPrEAsxMUsIFDZU (A Guide to Decision Trees for Machine Learning and Data Science)
- https://colab.research.google.com/drive/1q0Ppj2izj6yPuJ49rXkDxWxM-yLfly98#scrollTo=3CzEnsXm9_fx (Creating and Visualizing Decision Trees with Python)