|  |
| --- |
| PXL-Digital |
| Streamlit Applicaties |
| Streamlit applicaties runnen in notebook en deployen op Heroku |

|  |
| --- |
| Christiaan Prévot, Dennis Keusters, Kristof Heulsen  6-1-2020 |

# Doelstelling:

De bedoeling van deze opdracht is om verschillende notebook files aan te maken, die samenwerken met verschillende files (csv, pickle, images), Als alle files werken gaan we deze deployen op Heroku zodat ze online staan en je ze altijd kan raadplegen. Eens de apllicaties online staan kan je bekijken en er interactief mee kan werken.

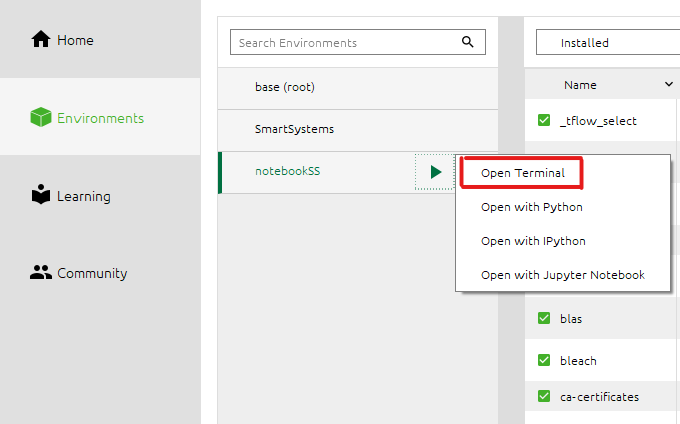
Verder beschrijft deze handleiding hoe men streamlit applicaties kan openen En hoe je zelf een eigen streamlit applicatie via notebook op heroku kan deployen. Ook word er uitgelegd hoe de zelfgemaakte streamlit applicaties werken en wat ze doen.

De code kan gedownload worden via de volgende link : https://github.com/ultratronics/smart\_systems\_eindoefening?fbclid=IwAR2Y\_BTy7itvyAxPCPPQBdXOBY4JYhO43q9HgJHpkZ5GafjNo\_9Ic2lcQvY

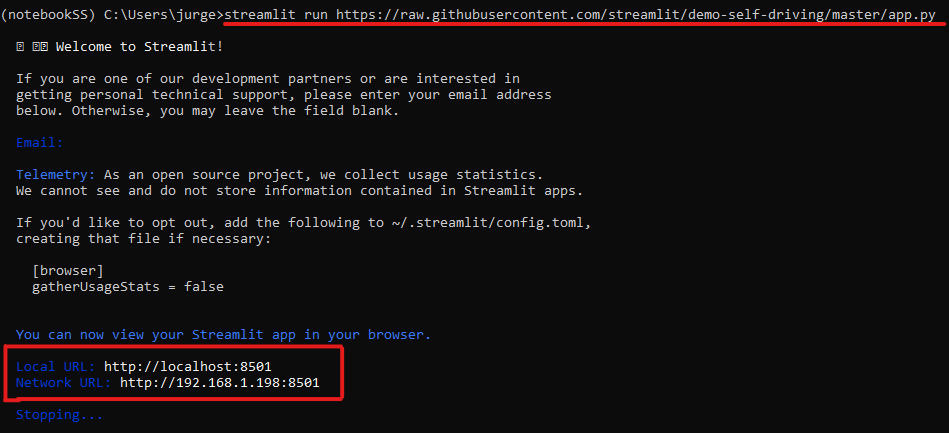
# Demo applicaties:

## Openen van streamlit applicatie demo Self-driving Car

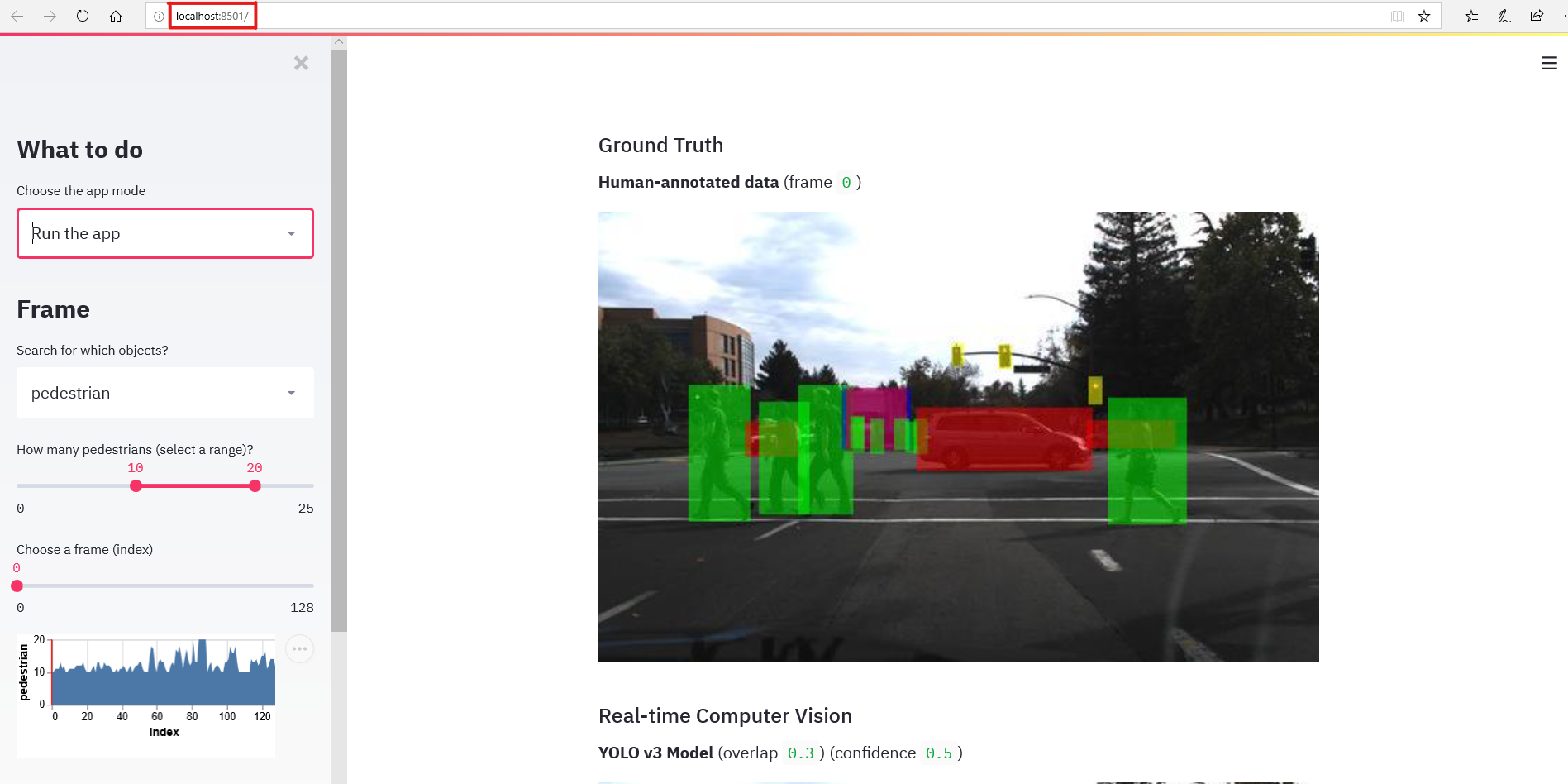
1. Open terminal.



1. Pip install Streamlit.
2. pip install --upgrade streamlit opencv-python.
3. streamlit run <https://raw.githubusercontent.com/streamlit/demo-self-driving/master/app.py>

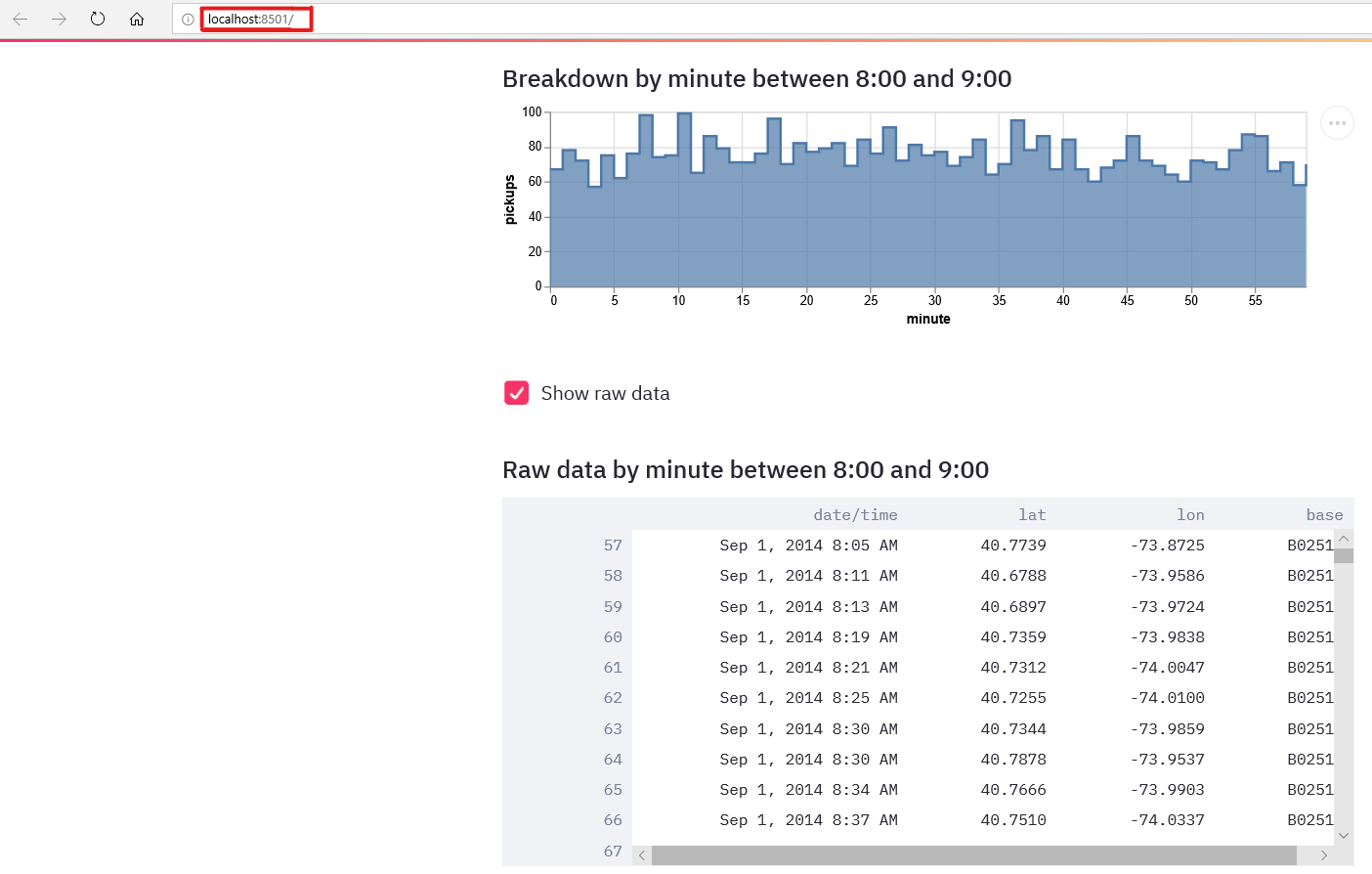


1. Normaal start de applicatie vanzelf anders vind je deze in de opgegeven ip-adres.
2. De applicatie gaat open in een internet tablad.
3. Links zie je de data inputs, dit zijn verschillende selectie-/ inputboxen en sliders die je kan veranderen om op de foto rechts andere dingen weer te geven. Dit zorgt voor een directe aanpassing die zichtbaar is op je app.



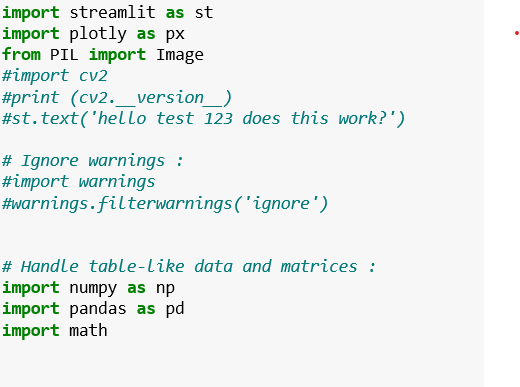
## Openen van streamlit applicatie demo uber drives

1. Open terminal (zie vorige).
2. pip install --upgrade streamlit (indien er een nieuwe versie is, anders ga je naar stap 3).
3. streamlit run <https://raw.githubusercontent.com/streamlit/demo-uber-nyc-pickups/master/app.py>

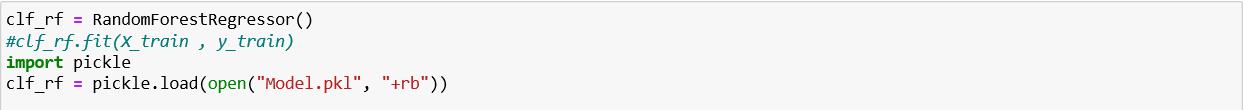


# Deploy een streamlit applicatie op Heroku

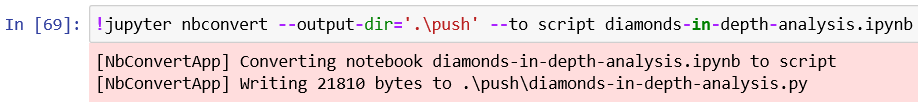
1. Open een jupyter notebook.
2. Importeren van de juiste libraries.

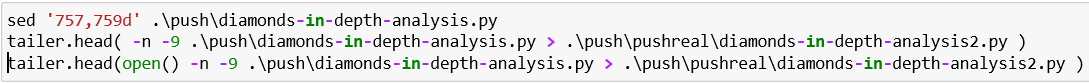


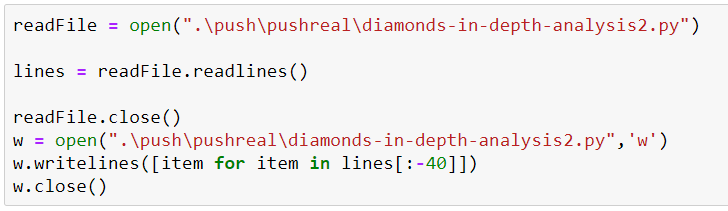
1. Inlezen van een pickle in notebook.



1. Converteren de notebook file naar een python file en sla het op een gekende plaats op.



1. Snij de laatste commando’s weg uit de file.
   1. Linux commando’s: Deze werken dus niet standaard op windows. Vervolgens maken we een copy van deze file en slaan hem op in een andere file die dan naar Heroku wordt gepushd.
   2. Python commando’s: Deze werken zowel in windows als in linux.



1. Push nu de hele map naar heroku via git commit en git push (je kunt dit best twee maal achter elkaar doen want soms vindt hij bij de eerste push geen veranderingen).

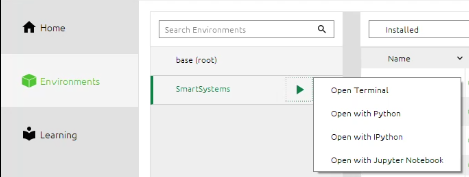


# Uitleg eigen applicaties:

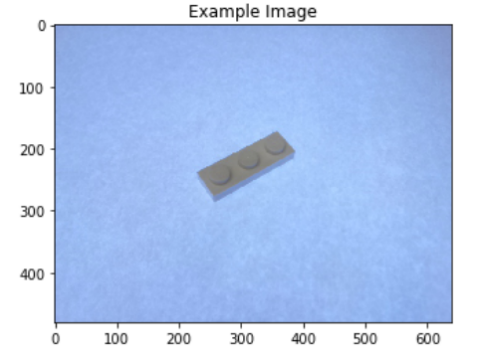
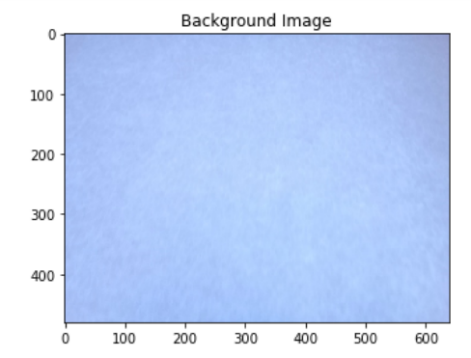
## Lego Dataset

Het doel van deze notebook is om verschillende soorten lego blokken te herkennen aan de hand van verschillende foto’s. Dit gaan we doen door middel van OpenCV.

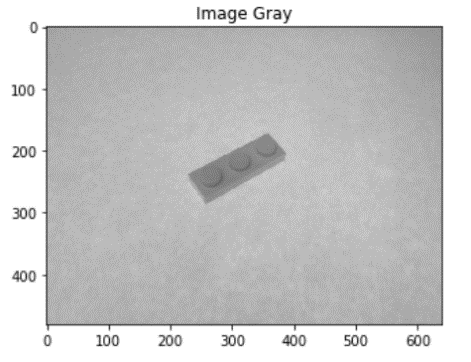
1. Open anaconda
2. Open uw environment
3. Open met jupyter notebook



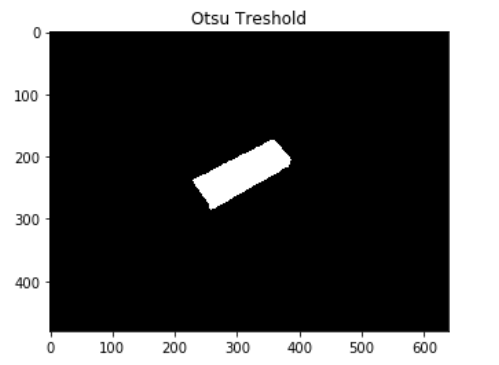
1. De notebook gaat open in een internet tablad.
2. Selecteer de map en open de python file.
3. Run de code met .
4. Als de applicatie klaar is.
   1. Zie je afbeeldingen van bepaalde lego blokken en van de achtergrond.



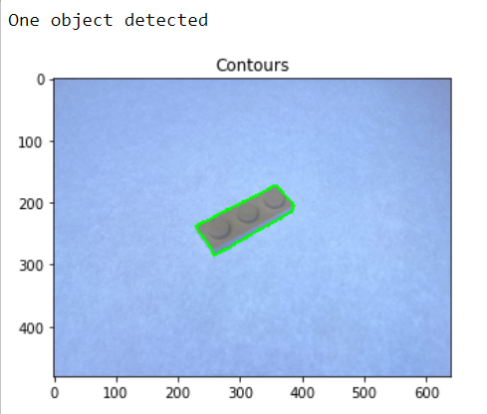
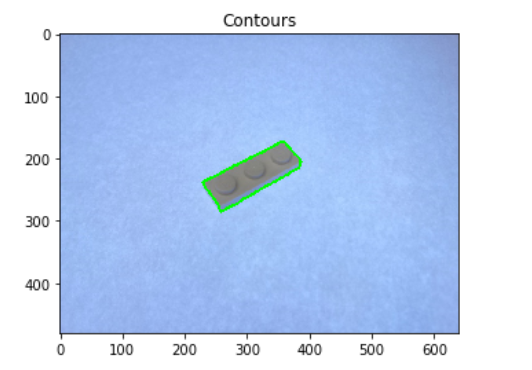
* 1. Vervolgens worden de afbeeldingen in Grayscale gezet.



* 1. De achtergrond word zwart en het object zelf word wit gemaakt zodat het onderscheid tussen object en achtergrond groot is.



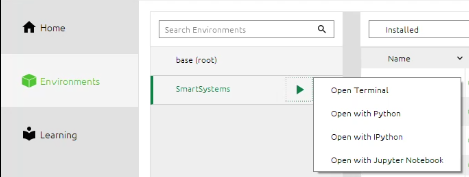
* 1. Er word een contour gemaakt rond het object dit is handig als er meerdere objecten zijn om ze zo gemakkelijk uit elkaar te halen en om te weten hoeveel objecten er zijn.



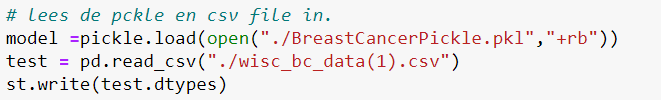
## Automatische Pickle file

In deze applicatie gaan we verschillende csv en pickle files online zetten op heroku, zodat je deze kan gaan gebruiken als streamlit applicatie.

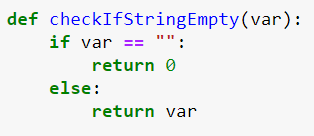
1. Open anaconda
2. Open uw environment
3. Open met jupyter notebook



1. De notebook gaat open in een internet tablad.
2. Selecteer de map en open de python file.
3. Run de code met .
   1. Er worden files ingelezen (csv en pickle)



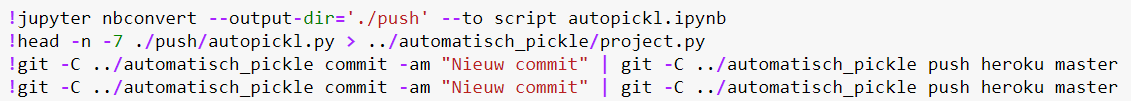
* 1. Een functie controleert of er geen lege string gebruikt wordt.



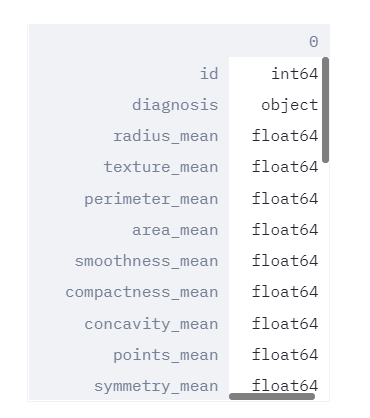
* 1. We doorlopen alle kolommen die er zijn in de csv file.



* 1. We kennen aan elke kolom toe of het een ingang of een uitgang is.
  2. Daarna converteren we de notebook file naar een python file.
     1. We verwijderen de laatste regels omdat deze na het runnen niet meer uitgevoerd mogen worden.
  3. We pushen de python file daarna naar heroku, waardoor er een streamlit applicatie aangemaakt wordt die je kan raadplegen.

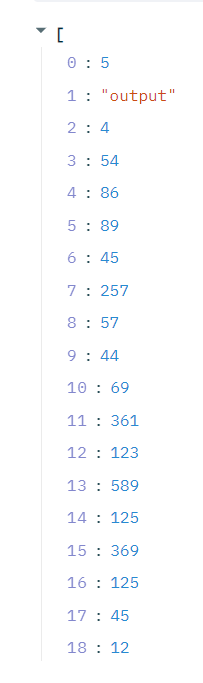


1. Als de applicatie klaar is kan je applicatie openen.
   1. Je kan de kolommen zien en welke type elke kolom is.



* 1. Je kan alle inputs een waarde meegeven en zo een eigen output waarde berekenen.

Input:

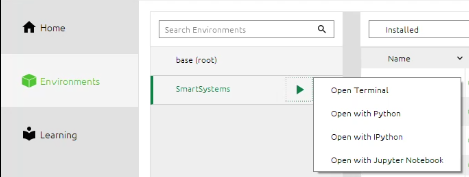
Output:



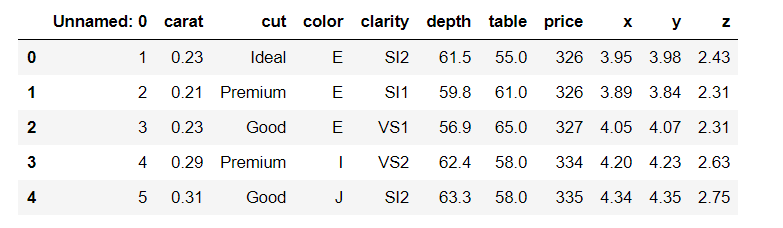
## Diamond

In deze applicatie gaan we een notebook maken waarin we bepaalde details van diamanten gaan bereken, ook kan deze data visueel worden weergegeven op grafieken en heatmaps.

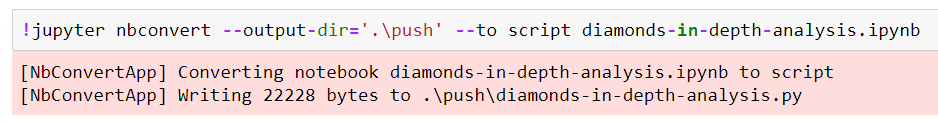
1. Open anaconda
2. Open uw environment
3. Open met jupyter notebook



1. De notebook gaat open in een internet tablad.
2. Selecteer de map en open de python file.
3. Run de code met .
4. De juiste libraries en csv file worden ingelezen.
5. We lezen de eerste 5 rijen in om te kijken wat er allemaal in de csv file zit.

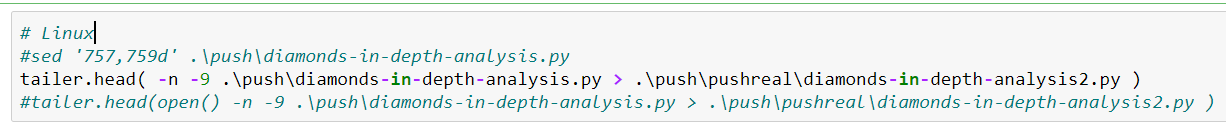


1. Daarna deleten we de Unnamed:0 kolom omdat we al een index kolom in het begin hebben.
2. Dan laten we verschillende grafieken zien in de notebook als visualisatie
3. De file wordt geconverteerd naar een python file.



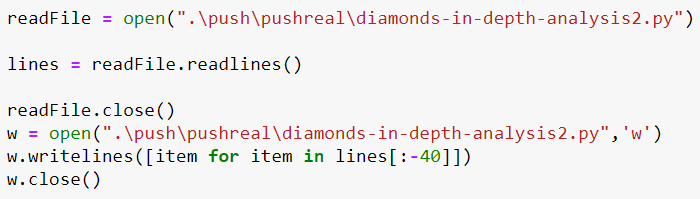
1. Verwijderen we de laatste regels van de file omdat hierin opnieuw de convert en de push naar heroku staan en dat moet hier niet meer.
2. Linux:

De file word bewerkt en automatich naar de nieuwe locatie gezet.



1. Python:

Enkel de laatste lijnen worden verwijdert.



1. Als je geen van beide doet moet je file zelf kopiëren naar de juiste locatie en vervolgens bewerken om de laatste regels die Ipython() beat te verwijderen.
2. Als laatste wordt de python file gepusht naar Heroku om zo de streamlit applicatie te runnen. Deze regel moet 2 keer uitgevoerd worden omdat hij soms bij de eerste push niet altijd de veranderingen waarneemt.

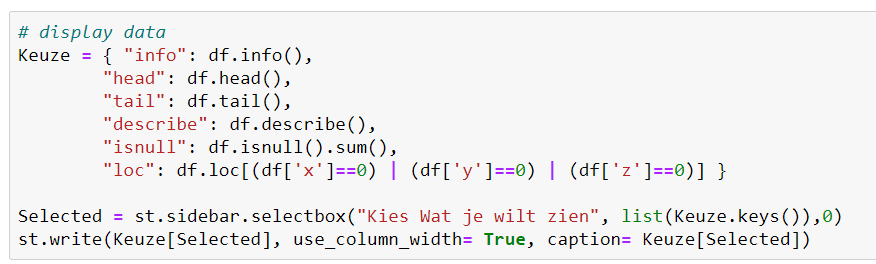


### De streamlit code in de notebook

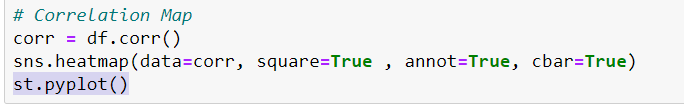
1. Eerst geven we een titel mee voor onze pagina.



1. Maken een selectbox met alle kolom inputs om zo de data te tonen



1. Als je standaard grafieken van de notebook file wilt displayen op de applicatie moet je er gewoon “st.pyplot() ” achter zetten. Dit werkt voor elke grafiek en heatmap.

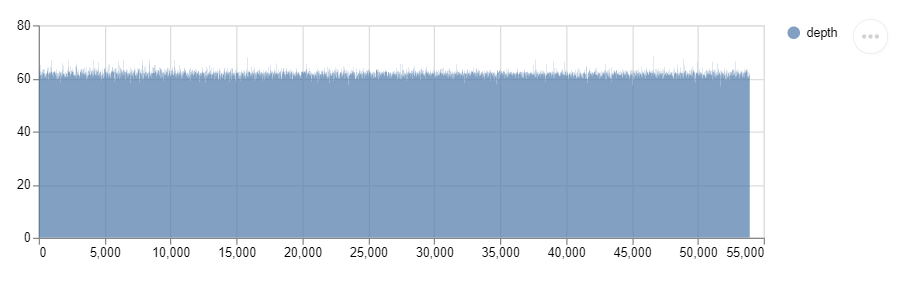


1. Je kan ook select boxen aanmaken om bepaalde elementen weer te geven op grafieken of tabellen.



Output:

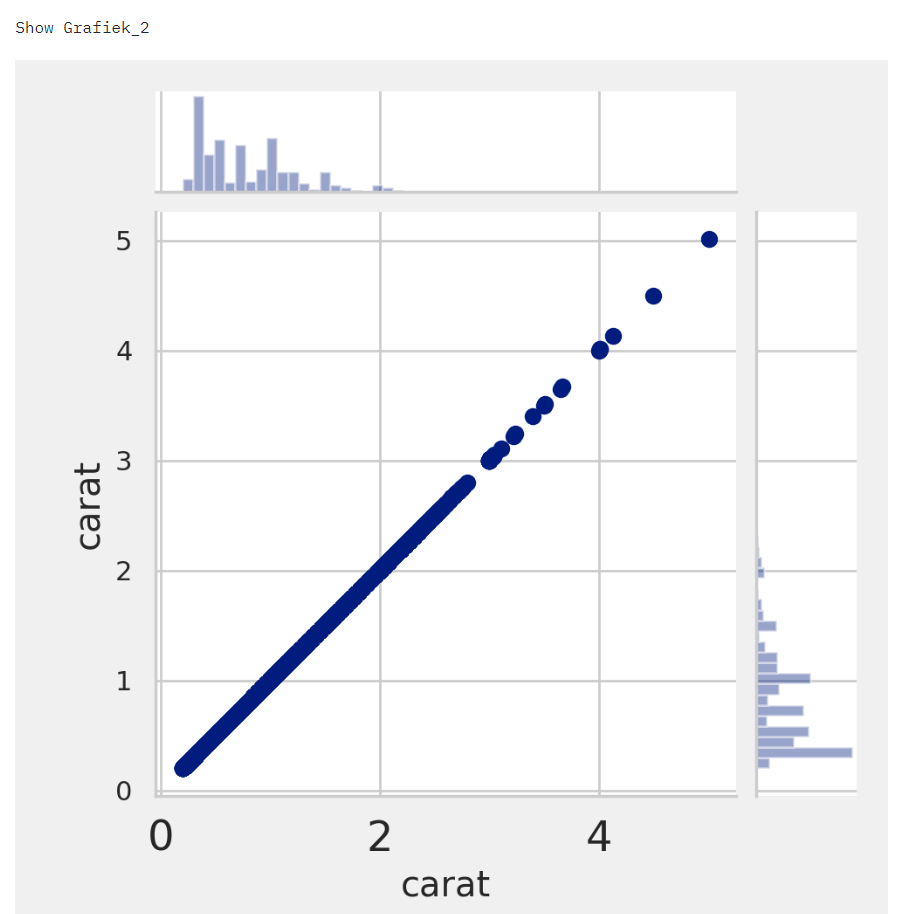
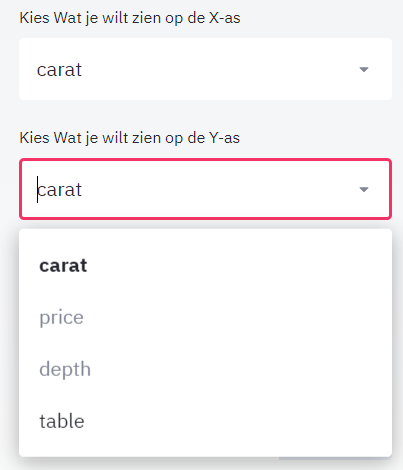




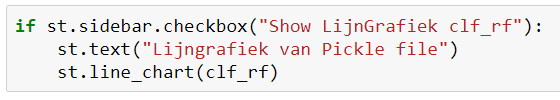
1. Met deze select boxen kan je ook de X- en Y-as naar keuze veranderen om verschillende grafieken te bekomen.



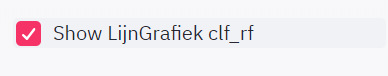
Output:

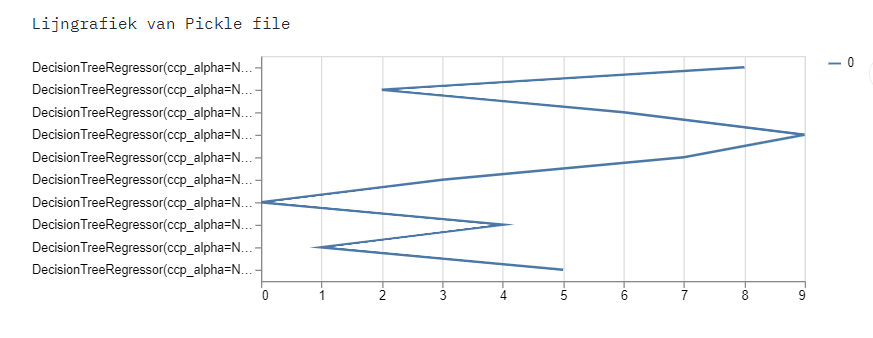


1. Als de applicatie te groot wordt kan je best gebruik maken van check boxen, dit betekend dat zolang de check box uit is deze niet getoond word in de applicatie

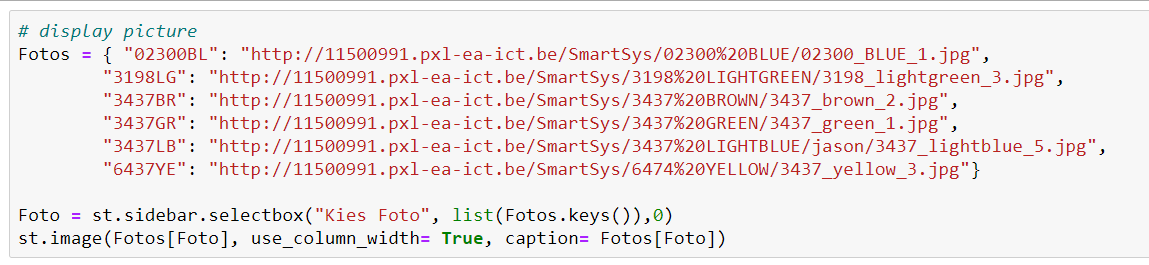


Output:





1. Er is ook een mogelijkheid om afbeeldingen weer te geven in streamlit, alleen moet je er dan wel voor zorgen dat de afbellding online op een server staan want anders gaat hij ze niet kunnen weergeven. (we hebben dit getest met de lego blokken ).
2. Een selectbox om een foto te kiezen.



1. Dan kan je nog code schrijven als er een bepaalde foto geselecteerd is.



Output:





## Opmerkingen

1. Numbers als input kunnen voor errors zorgen