Verslag Project Musti Groep TIAO2

**Groepsleden:** - Bart De Paepe

- Berthold Dewaele

- Tomas Boone

- Wesley Maebe

- Ellen De Smedt

# Het verkennen van de data

Veel verkenning was er voor dit project niet, de structuur van de data was zeer duidelijk en de aangeboden foto's moesten geclassificeerd worden binnen de 3 opgegeven klassen. Wel werd er geëxperimenteerd met het omzetten van een willekeurige foto naar bruikbare datatypes, het effect van het spelen met de rgb-waarden van de foto’s en, omdat we nog maar net aan de cursus begonnen, werd er ook eerst even geëxperimenteerd met binaire classificatie tussen 2 van de 3 mogelijke klassen.

# Het voorbereiden van de data

Hier moesten we uitzoeken hoe we een reeks foto’s konden omvormen tot data die gebruikt kon worden om een model te trainen. Nadien moest deze data terug omgezet kunnen worden naar een gewone foto die door de gebruiker bekeken kan worden. Daarbij zijn we op zoek gegaan naar de meest geschikte grootte of het meest geschikte aantal pixels om de verwerkingssnelheid binnen een redelijke termijn te houden, maar tegelijk niet te veel detail te verliezen. De foto’s werden uiteindelijk omgezet naar een numpy array en op basis daarvan werd het dataframe aangemaakt. Daarna werd de data opgesplitst in een test-set en een trainings-set. We passen ‘scaling’ toe op de dataset omdat de geselecteerde modellen beter presteren met geschaalde data.

# Het uittesten van de modellen

Verscheidene modellen werden uitgeprobeerd: De Stochastic Gradient Descent Classifier (SGD), de One Vs One Classifier, de Random Forest Classifier, de Lineair Support Vector Classifier (SVC) en tot slot de Voting Classifier. Van al deze modellen bleek de Random Forest Classifier het best te presteren met een initiële accuraatheid van iets meer dan 94%. Bijgevolg hebben we ervoor geopteerd om met de Random Forest Classifier verder te gaan.

# Fine-tuning en error-analyse

In een poging om de prestaties van het model te verbeteren zijn we op zoek gegaan naar de mogelijkheden die geboden worden door het aanpassen van de parameters van de Random Forest Classifier. Hierbij hadden we vooral interesse in de parameters die invloed kunnen hebben op de accuraatheid zoals bv. n\_estimators of max\_depth, maar ook in max\_leaf\_nodes, een parameter die kan helpen om over-fitting tegen te gaan door de max\_depth te helpen beperken. Vervolgens gingen we op zoek naar de beste instellingen op basis van random search en grid search.

Voor de error-analyse maakten we gebruik van de confusion matrix die berekend werd op het gekozen model en nadien op het gewogen model. Om dit laatste model te bekomen werd er m.b.v. de parameter class\_weight gewicht toegevoegd aan de 2e klasse van het model. Tot slot werd er geëxperimenteerd met de threshold om te bepalen welke waarde hieraan toegekend zou worden.

# De applicatie

Bij de app hebben we gekozen voor eenvoud. Het is een flask applicatie geschreven in Python die de gebruiker een foto toont met de gevonden classificatie en de tijdsgegevens voor de foto. Daarnaast is er ook de mogelijkheid om het model te hertrainen door een simpele druk op de knop. Voor het tonen van de foto start de applicatie van de meest recent genomen foto en loopt het terug naar de eerste foto waarop een kat aanwezig is volgens het model en waarvoor de threshold behaald werd, m.a.w. het negeert de foto's die geclassificeerd worden als "geen kat aanwezig" of foto's die niet met voldoende zekerheid geclassificeerd konden worden.