Wie funktioniert das System ?

Das Firenet-System besteht im Wesentlichen aus zwei Teilen: der Standortanalyse und der Einheit selbst.
Im ersten Teil wird gezeigt, wie sich unser Modell für eine komparative Analyse verschiedener Standorte nutzen lässt.

Standortanalyse

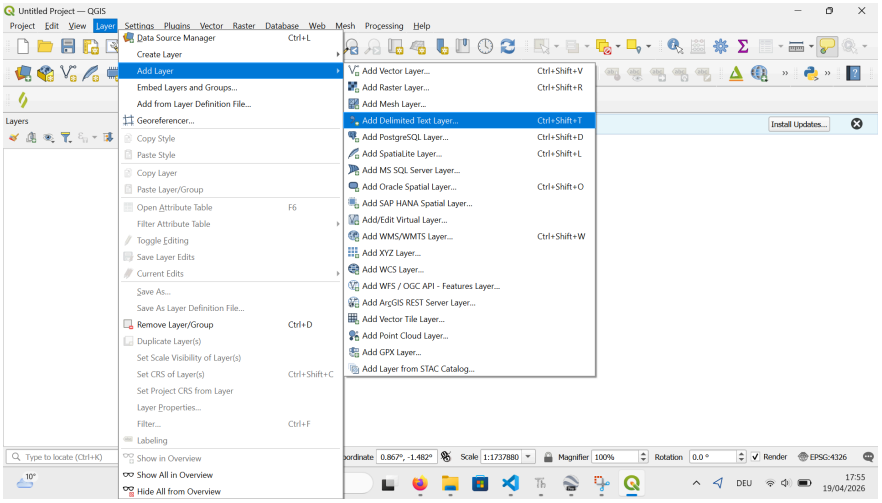
Um die Standortanalyse durchzuführen, ist es am einfachsten, die Open-Source-Software QGIS (https://qgis.org/) in Kombination mit Pythons scikit-learn (https://scikit-learn.org) zu verwenden. Für Python empfehlen wir, die Benutzeroberfläche von Google Colab (https://colab.research.google.com/) zu nutzen.

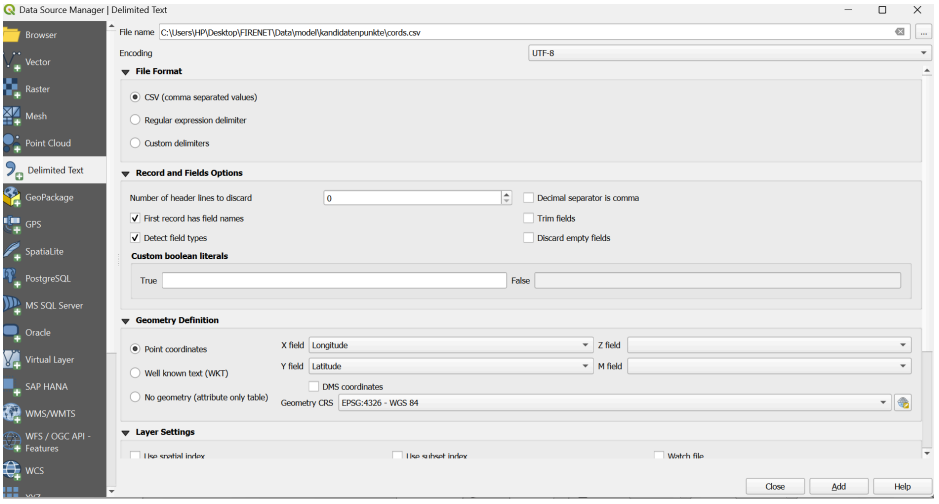
Zunächst muss unser Modell über GitHub heruntergeladen werden (https://github.com/FIRENET-AI/FIRENET-AI.github.io/main/code/model_2.pkl) .
Nachdem das Modell heruntergeladen wurde, braucht man verschiedene Standorte, die es zu vergleichen gilt. Diese lassen sich über Google Maps bestimmen und deren Koordinaten exportieren.

Die Koordinaten nimmt man am einfachsten im CSV-Format (lat, lon) auf.

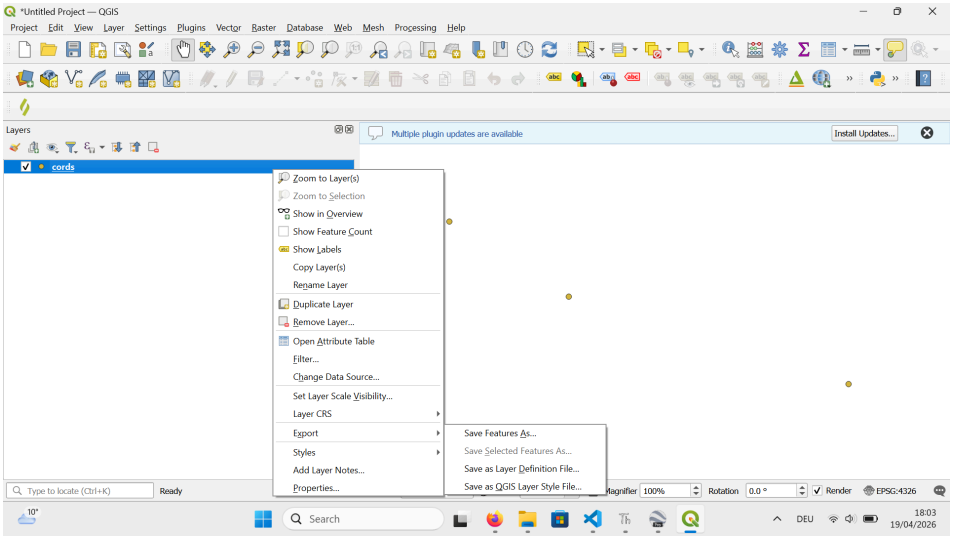
Diese CSV-Datei muss nun in QGIS geladen werden. Dafür geht man über Layer > Add Layer > Add Delimited Text Layer und setzt Geometry CRS auf

EPSG 4326 (WGS 84).

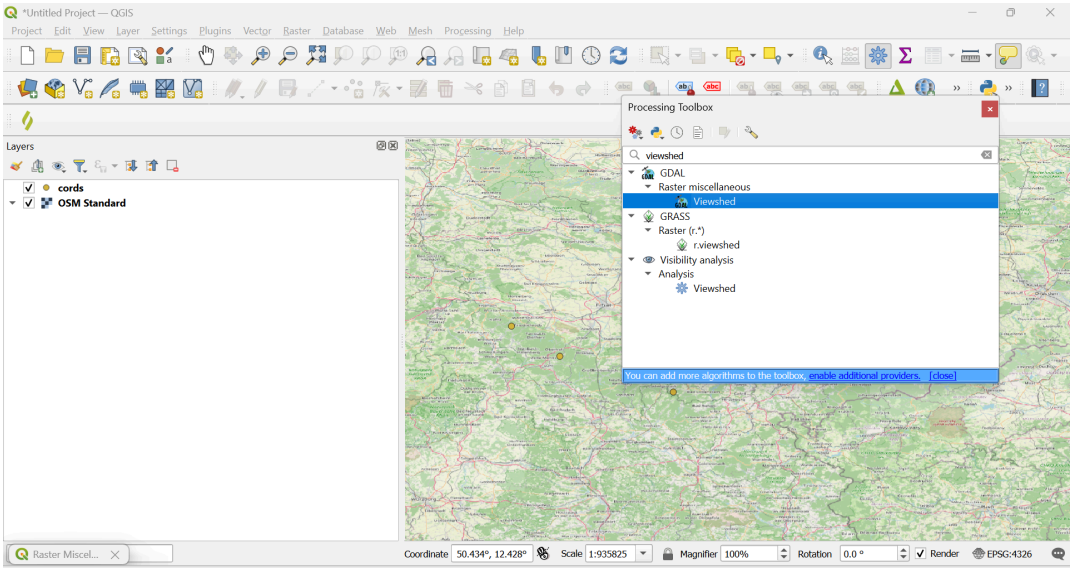




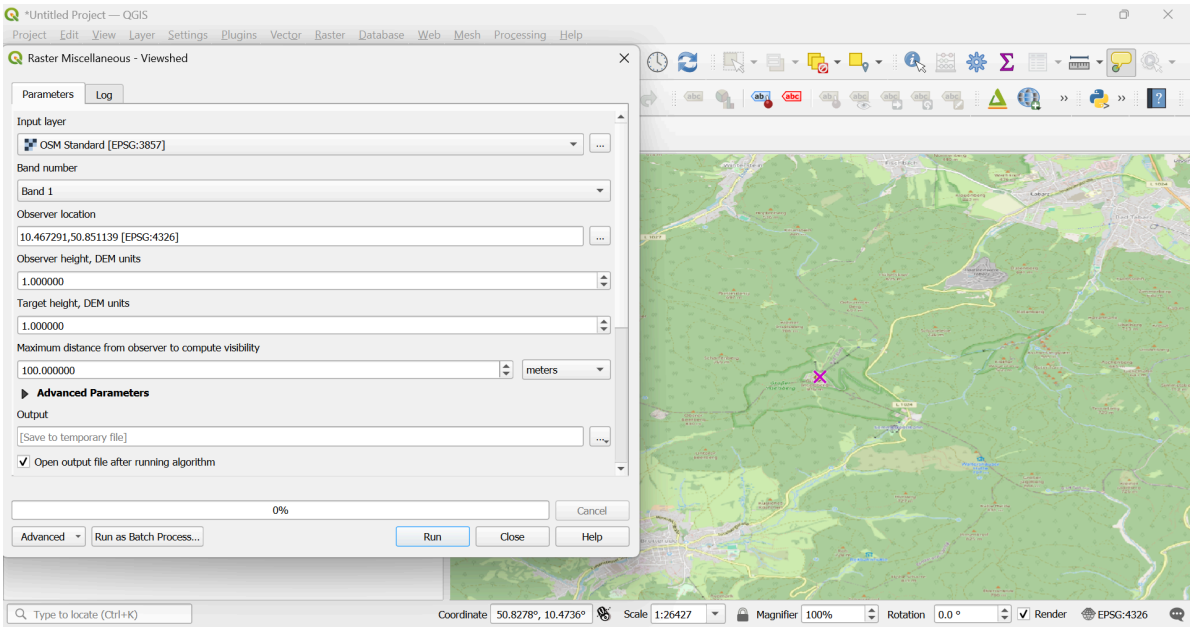
Anschließend projiziert man diese Vektordaten um – dazu exportiert man den Layer als GeoJSON in EPSG 25832.

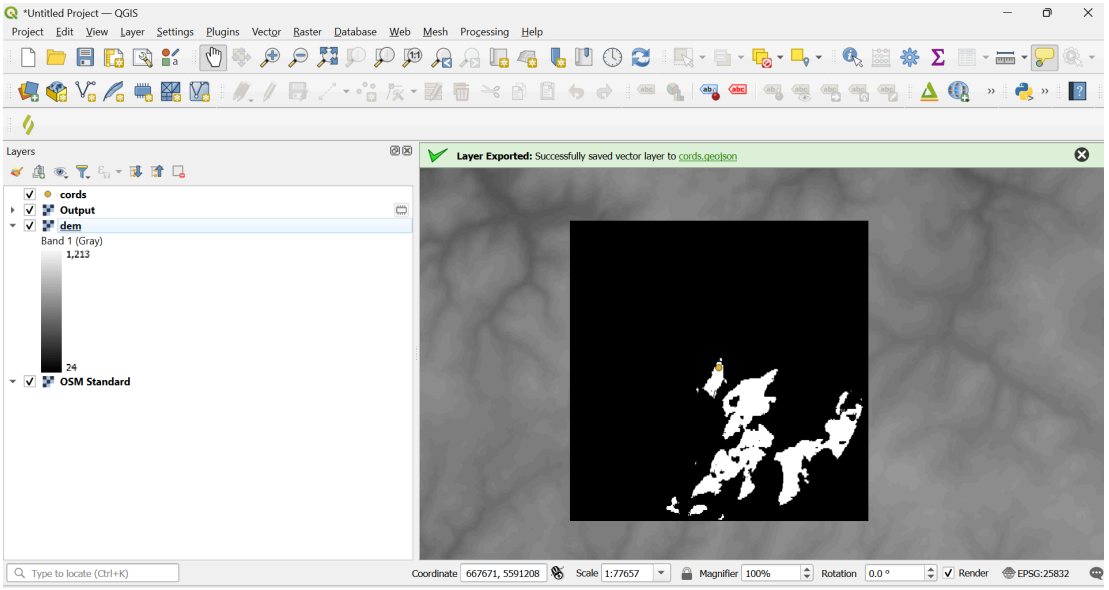


Jetzt wird die Viewshed-Analyse durchgeführt. Dafür braucht man ein DSM (Digital Surface Model), das sich am einfachsten über OpenTopography herunterladen lässt (https://portal.opentopography.org). Die Viewshed-Analyse gelingt über GDALs Viewshed-Tool.

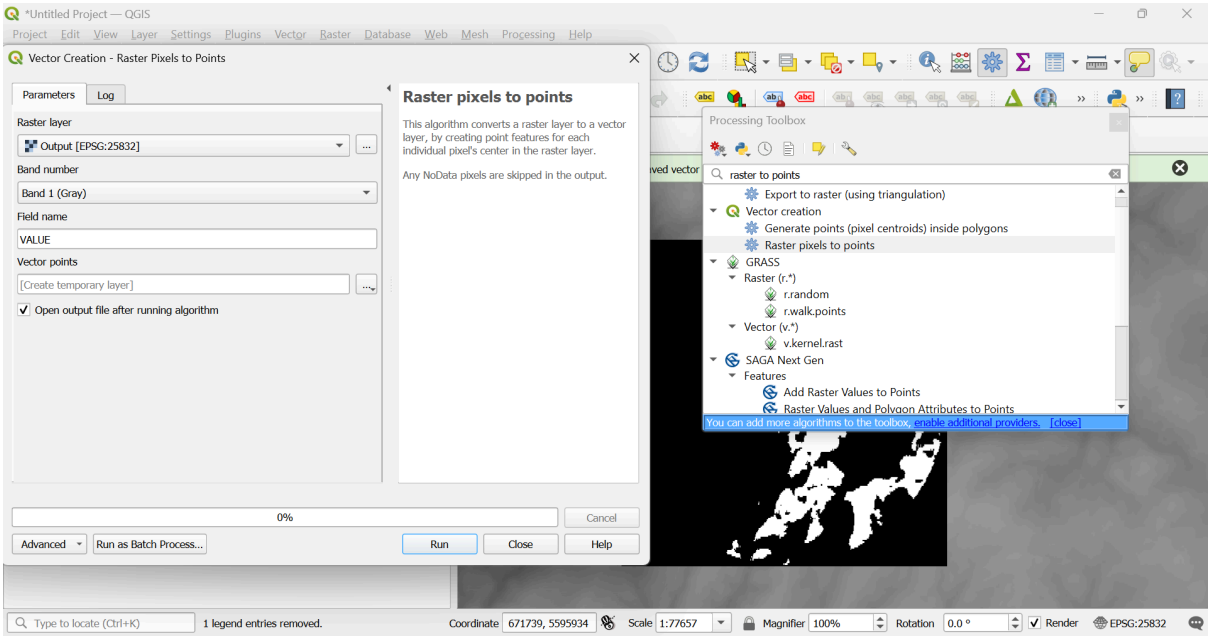


`max_dist` wird auf 3500 m gesetzt. `observer_height` bleibt null, wenn man ein DSM nutzt und das Gelände darauf abgebildet ist. Nutzt man DEM, wird die Höhe des Punktes über Grund genommen ; dann wird die Viewshed durchgeführt.

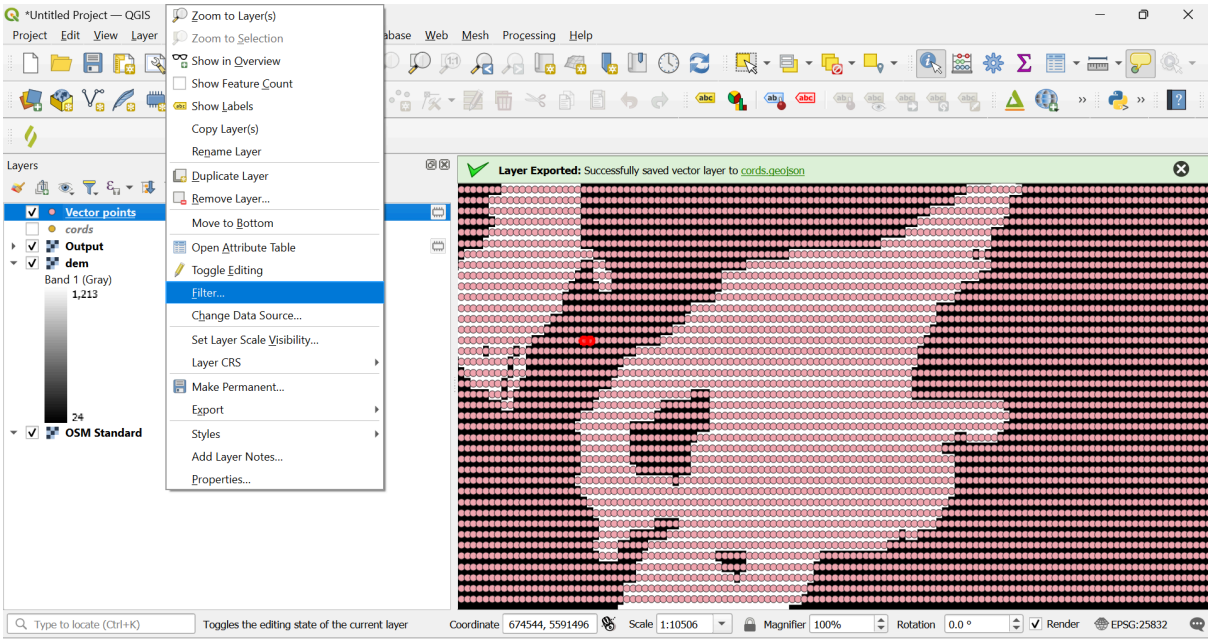


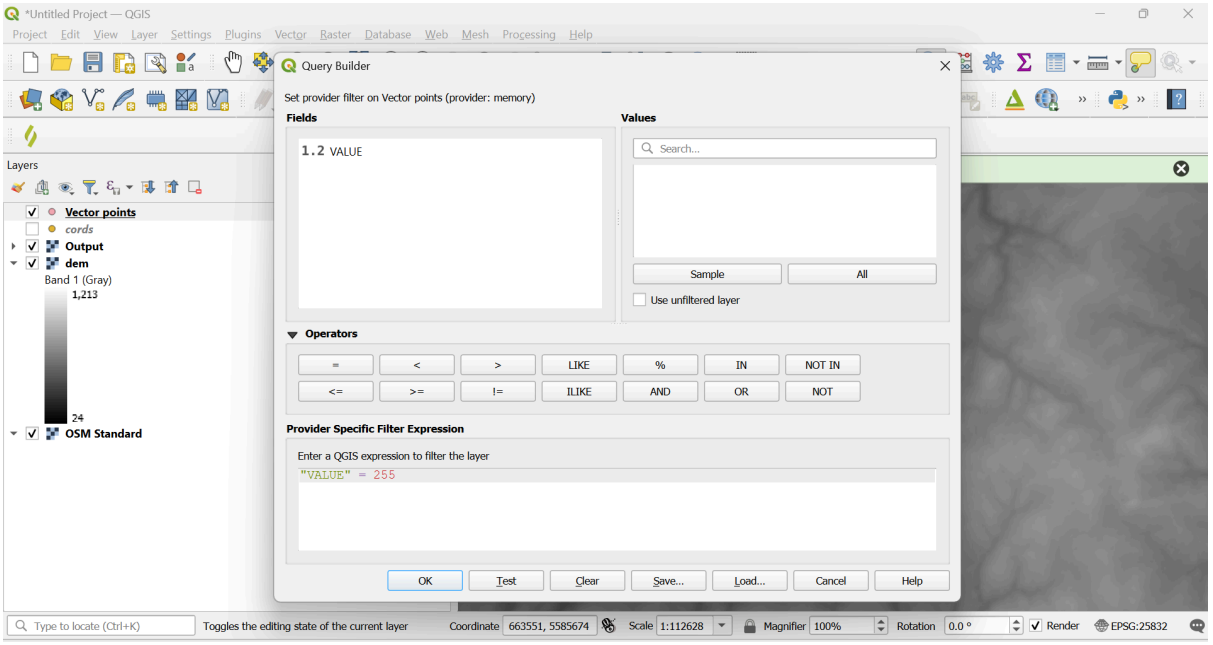


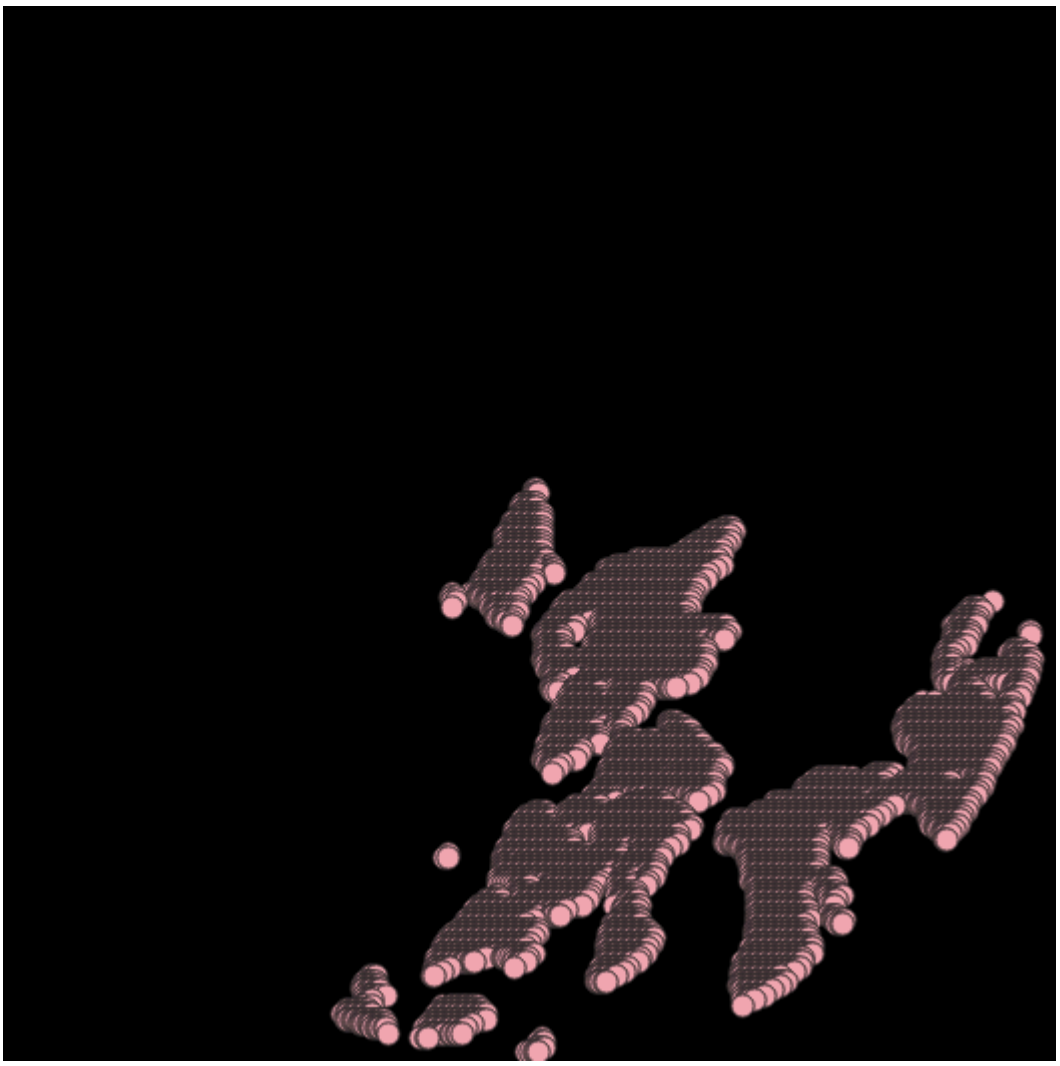
Dieses Rasterformat wird nun über "Raster pixels to points" in ein Vektorformat umgewandelt.



Diese Points werden anschließend auf die sichtbaren Punkte gefiltert.

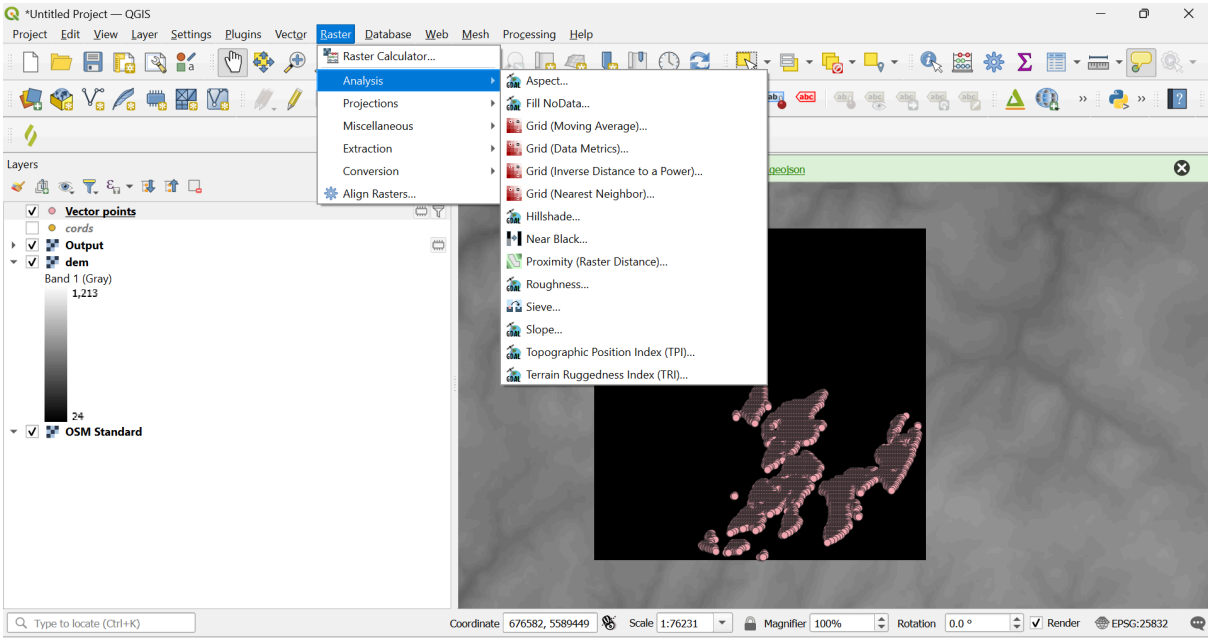

*Untitled Project — QGIS
Project Edit View Layer
Zoom to Layer(s)
Zoom to Selection
Show in Overview
Show Feature Count
Show Labels
Copy Layer(s)
Rename Layer
Duplicate Layer
Remove Layer...
Move to Bottom
Open Attribute Table
Toggle Editing
Filter...
Change Data Source...
Set Layer Scale Visibility...
Layer CRS
Make Permanent...
Export
Styles
Add Layer Notes...
Properties...
Layers
Vector points
cords
Output
dem
Band 1 (Gray)
1,213
24
OSM Standard
Layer Exported: Successfully saved vector layer to cords.geojson
Type to locate (Ctrl+K)
Toggles the editing state of the current layer
Coordinate 674544, 5591496
Scale 1:10506
Magnifier 100%
Rotation 0.0 °
Render
EPSG:25832


*Untitled Project — QGIS
Query Builder
Set provider filter on Vector points (provider: memory)
Fields
1.2 VALUE
Values
Search...
Sample
All
Use unfiltered layer
Operators
= < > LIKE % IN NOT IN
<= >= != ILIKE AND OR NOT
Provider Specific Filter Expression
Enter a QGIS expression to filter the layer
"VALUE" = 255
OK
Test
Clear
Save...
Load...
Cancel
Help
Layers
Vector points
cords
Output
dem
Band 1 (Gray)
1,213
24
OSM Standard
Type to locate (Ctrl+K)
Toggles the editing state of the current layer
Coordinate 663551, 5585674
Scale 1:112628
Magnifier 100%
Rotation 0.0 °
Render
EPSG:25832



Für diese sichtbaren Punkte müssen nun folgende Parameter aufgenommen werden: `['buildings_m', 'roads_m', 'recreation_ground_m', 'residential_m', 'campsites_m', 'dem', 'tri', 'tcd', 'dist_forest_edge']`

`buildings_m, roads_m, recreation_ground_m, residential_m und campsites_m` können über Geofabrik für den Standort heruntergeladen werden (https://download.geofabrik.de/). Diese Vektordaten werden wiederum über "Raster conversion > Vector to raster" konvertiert und eine Proximity Map wird über "Raster research > Proximity map (raster distance)" erstellt.

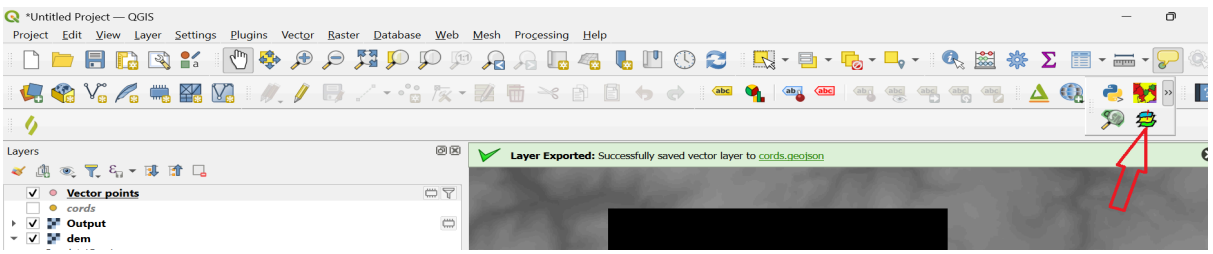


Für die topografischen Features lassen sich alle relevanten Features über das geladene DEM berechnen – dafür geht man über "Research & Analyse" und berechnet TRI.

Die Tree Cover Density (tcd) lässt sich über Copernicus herunterladen: https://land.copernicus.eu/en/products/high-resolution-layer-forests-and-tree-cover?tab=tree_cover_density

Für `dist_forest_edge` wird zunächst jede Zelle mit tcd ≤ 50 % als 0 und darüber als 1 gesetzt. Auf diese Maske wird ebenfalls das Proximity-Tool angewendet.

Nun hat man 9 Feature-Layers. Diese Layers können per Drag & Drop wieder in QGIS geladen werden und für jeden Punkt über das Point Sampling der sichtbaren Punkte angehängt werden.



Diese Punkte können als CSV exportiert werden. Basierend auf dieser CSV und dem Modell kann nun die Analyse mit dem folgenden Script durchgeführt werden:

```
import pandas as pd
import pickle


fire = pd.read_csv("koordinaten.csv")

features = ['buildings_m', 'roads_m',
'recreation_ground_m',
            'residential_m', 'campsites_m', 'dem', 'tri',
            'tcd', 'dist_forest_edge']

X = fire[features]


with open("model.pkl", "rb") as f:
    model = pickle.load(f)

proba = model.predict_proba(X)
fire["pred_proba"] = proba[:, 1]

fire.loc[fire["tcd"] < 50, "pred_proba"] = 0

sum_pred = fire["pred_proba"].sum()

print("Summe der vorhergesagten Wahrscheinlichkeiten:",
sum_pred)

fire.to_csv("with_predictions.csv", index=False)
```

PS: Wir arbeiten gerade an einer Softwarelösung, die den gesamten Prozess wesentlich vereinfachen wird.