



# Biodiversity Capstone Project

INVESTIGATING PROTECTED SPECIES

PRESENTED BY MARC HANDS

# Species\_info.csv

The csv file contains information about wildlife species and their conservation status. The columns in the table were the following :

- Category : mammals, bird, etc.
- Scientific name
- Common names
- Conservation status

The object of this presentation is to analyze different specie's conservation status using Python.

Read csv file using pandas. Save dataframe as variable “species”.

```
species = pd.read_csv('species_info.csv')
```

# Print(species.head())

Figure 1 : species

	category	scientific_name	common_names	conservation_status
0	Mammal	Clethrionomys gapperi gapperi	Gapper's Red-Backed Vole	nan
1	Mammal	Bos bison	American Bison, Bison	nan
2	Mammal	Bos taurus	Aurochs, Aurochs, Domestic Cattle (Feral), Domesticated Cattle	nan
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	nan
4	Mammal	Cervus elaphus	Wapiti Or Elk	nan
5	Mammal	Odocoileus virginianus	White-Tailed Deer	nan
6	Mammal	Sus scrofa	Feral Hog, Wild Pig	nan
7	Mammal	Canis latrans	Coyote	Species of Concern
8	Mammal	Canis lupus	Gray Wolf	Endangered
9	Mammal	Canis rufus	Red Wolf	Endangered

Calculate the number of unique species :

```
species_count = species['scientific_name'].nunique()
print(species_count)
>> 5541
```

What are the unique entries in the category column ?

```
species_type = species['category'].unique()
print(species_type)
>> ['Mammal' 'Bird' 'Reptile' 'Amphibian' 'Fish' 'Vascular Plant' 'Nonvascular Plant']
```

What are the different values of conservation\_status ? :

```
conservation_statuses = species['conservation_status'].unique()
print (conservation_statuses)
```

How many unique scientific names fall into each conservation category :

```
conservation_counts=species.groupby('conservation_status').scientific_name.nunique().reset_index()
```

>> Figure 2 : conservation\_counts :

	Conversation_status	Scientific_name
0	Endangered	15
1	In Recovery	4
2	Species of Concern	151
3	Threatened	10

What this table reveals is that less than 200 of the 5541 species in our dataframe are classified under a conservation status. All others are listed as “Nan”, or Not a number.

To account for species without a conservation status, we fill in NaN values with 'No Intervention' using :

```
species.fillna('No Intervention', inplace = True)
```

With another iteration of groupby, we get :

>> Figure 3 : conservation\_counts : “No Intervention” row added

	Conservation_status	Scientific_name
0	Endangered	15
1	In Recovery	4
2	No Intervention	5363
3	Species of Concern	151
4	Threatened	10

# Conservation Status by species

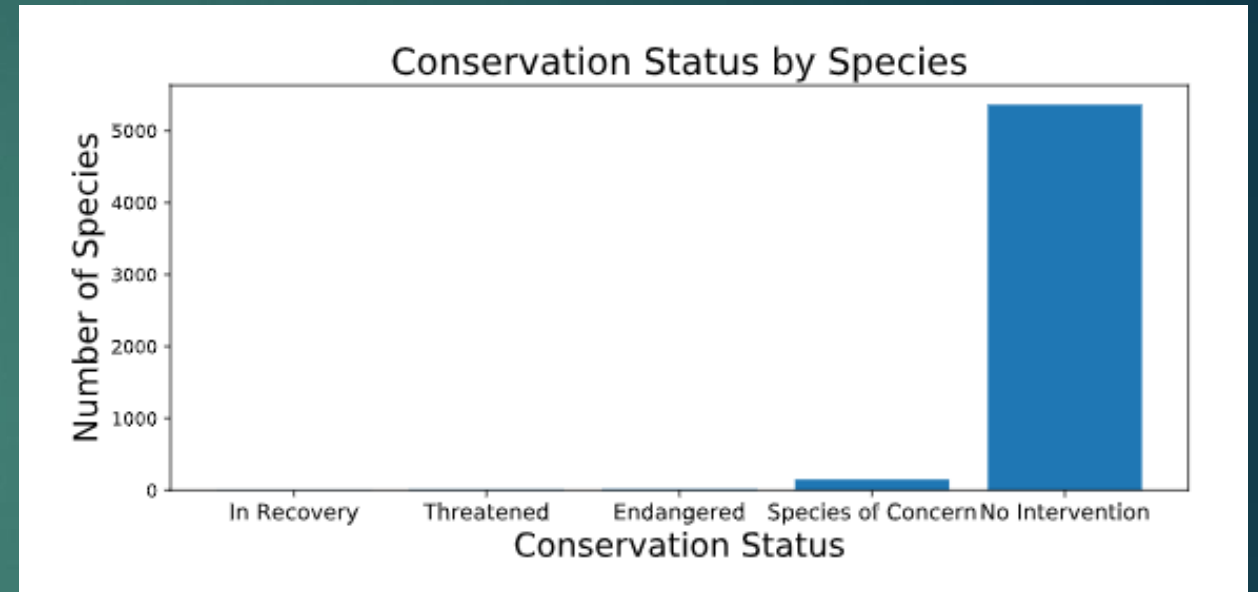
The following bar graph further illustrates how few species have a prescribed conservation status.

```
plt.figure(figsize=(10,4))
ax=plt.subplot()
plt.bar(range(len(protection_counts['conservation_status'])),
        protection_counts['scientific_name'])

ax.set_xticks(range(len(protection_counts.conservation_status)))
ax.set_xticklabels(protection_counts.conservation_status,
                   fontsize=12)
plt.ylabel('Number of Species', fontsize = 18)
plt.xlabel('Conservation Status', fontsize = 18)
plt.title('Conservation Status by Species', fontsize = 20)

plt.show()
```

Figure 4 :



# Investigating endangered species

- ▶ In the following exercise, we try to see if certain species are more likely to be endangered than others. By creating a new column “is\_protected” that holds false for all species with “No Intervention” and True otherwise.
  - ▶ `species['is_protected'] = species.conservations_status != 'No Intervention'`
- ▶ Now grouping by category, and counting the number of species per category for True and False values of is\_protected.
  - ▶ `category_counts = species.groupby(['category', 'is_protected']).scientific_name.nunique().reset_index()`
- ▶ There is a large number of vascular plants that don't have any intervention. Maybe they should be an area of focus for scientists.

	category	is_protected	scientific_name
0	Amphibian	False	72
1	Amphibian	True	7
2	Bird	False	413
3	Bird	True	75
4	Fish	False	115
5	Fish	True	11
6	Mammal	False	146
7	Mammal	True	30
8	Nonvascular Plant	False	328
9	Nonvascular Plant	True	5
10	Reptile	False	73
11	Reptile	True	5
12	Vascular Plant	False	4216
13	Vascular Plant	True	46

Figure 5 : category\_counts



# Pivoting tables

Although the previous table has all the information we need, it's hard to read and a little redundant. Pivoting the table to get a column for “protected” and “not\_protected” organizes the information for a single species all a single row.

```
category_pivot =  
category_counts.pivot(columns='is_protected',  
                        index='category',  
                        values='scientific_name')\  
.reset_index()
```

```
category_pivot.columns = ['category', 'not_protected',  
                          'protected']
```

Adding a column for percentage protected:

```
category_pivot['percent_protected'] =  
100.0*category_pivot.protected /  
(category_pivot.protected +  
category_pivot.not_protected)
```

Figure 6 : category\_pivot

Category pivot	not_protected	protected	Percent_protected
Amphibian	72	7	8.86
Bird	413	75	15.36
Fish	115	11	8.73
Mammal	146	30	17.05
Nonvascular Plant	328	5	1.5
Reptile	73	5	6.41
Vascular Plant	4216	46	1.08



# Chi-Squared Test

- In the previous table, we can see that we have categorical data. We want to verify the null hypothesis, which in this case, would be that a specie's category and its protection status are independent. We are comparing two categories at a time. A chi-squared test resulting in a pval smaller than 0.05 would indicate we can reject the null hypothesis.

Figure 7 : conservation status of mammals and birds

Category	Protected	Not-Protected	Pval
Mammal	30	146	0.6876
Bird	75	413	

Figure 8 : conservation status of reptiles and mammals

Category	Protected	Not-Protected	Pval
Reptile	5	73	0.0384
Mammal	30	146	

- While the high Pval of mammals vs birds indicates the null hypothesis is true, we can see from reptiles vs mammals that some species are more to be protected than others.

# In search of Sheep

We are given 2 dataframes : the previous species dataframe and a new “observations” dataframe, containing a species’ scientific name, where this species was sighted, and how many sightings at that location occurred. Since we can’t tell by it’s scientific name whether a species is a sheep or not, we’ll have to use the “common names” column in “species”.

Create a column in species populated with the help of a lambda function returning true if “common\_names” column contains “sheep”.

```
species['is_sheep'] = species.common_names.apply(lambda x: 'Sheep' in x)
```

Declare a new dataframe containing only rows with True values of “is\_sheep”.

```
species_is_sheep = species[species.is_sheep]
```

```
print species_is_sheep
```

Figure 9 : species\_is\_sheep

category	scientific_name	common_names	conservation_status	is_protected	is_sheep
Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
Vascular Plant	Rumex acetosella	Sheep Sorrel, Sheep Sorrell	No Intervention	False	True
Vascular Plant	Festuca filiformis	Fineleaf Sheep Fescue	No Intervention	False	True
Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True
Vascular Plant	Rumex acetosella	Common Sheep Sorrel, Field Sorrel, Red Sorrel, Sheep Sorrel	No Intervention	False	True
Vascular Plant	Rumex paucifolius	Alpine Sheep Sorrel, Fewleaved Dock, Meadow Dock	No Intervention	False	True
Vascular Plant	Carex illota	Sheep Sedge, Smallhead Sedge	No Intervention	False	True
Vascular Plant	Potentilla ovina var. ovina	Sheep Cinquefoil	No Intervention	False	True
Mammal	Ovis canadensis sierrae	Sierra Nevada Bighorn Sheep	Endangered	True	True

Some of these species are vascular plants. We’re only interested in mammals.

# In search of sheep

```
sheep_species = species[(species.is_sheep) & (species.category == 'Mammal')]
```

Figure 10 : sheep\_species

category	scientific_name	common_names	conservation_status	is_protected	is_sheep
Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True
Mammal	Ovis canadensis sierrae	Sierra Nevada Bighorn Sheep	Endangered	True	True

This table contains only mammal and sheep rows of “species”. Now that we have the scientific names to lookup in the observation table, we merge “sheep\_species” and “observations” using an outer merge. Since both dataframes have the scientific\_name column, the merge will only keep rows with matching scientific names, effectively only keeping sheep species, and getting rid of other scientific names from “observations that aren’t sheep.”

```
sheep_observations = pd.merge (sheep_species , observations, how='outer')
```

# Sheep Observations by Park

Although `sheep_observations` contains sightings of all species of sheep in each park, we are only interested in the total number of sheep by park, so grouping by park name and summing the number of observations will provide a global picture of how many sheep sightings we can expect in each park.

Figure 11 : `obs_by_park`

	<b>park_name</b>	<b>observations</b>
0	Bryce National Park	576025
1	Great Smoky Mountains National Park	431820
2	Yellowstone National Park	1443562
3	Yosemite National Park	863332

```
obs_by_park = sheep_observations.groupby('park_name').observations.sum().reset_index()
```

An additional `groupby` could have been done to evaluate the number of sightings per sheep species per park. Perhaps “*Ovis Aerus*” could have significantly more sightings in a single park?

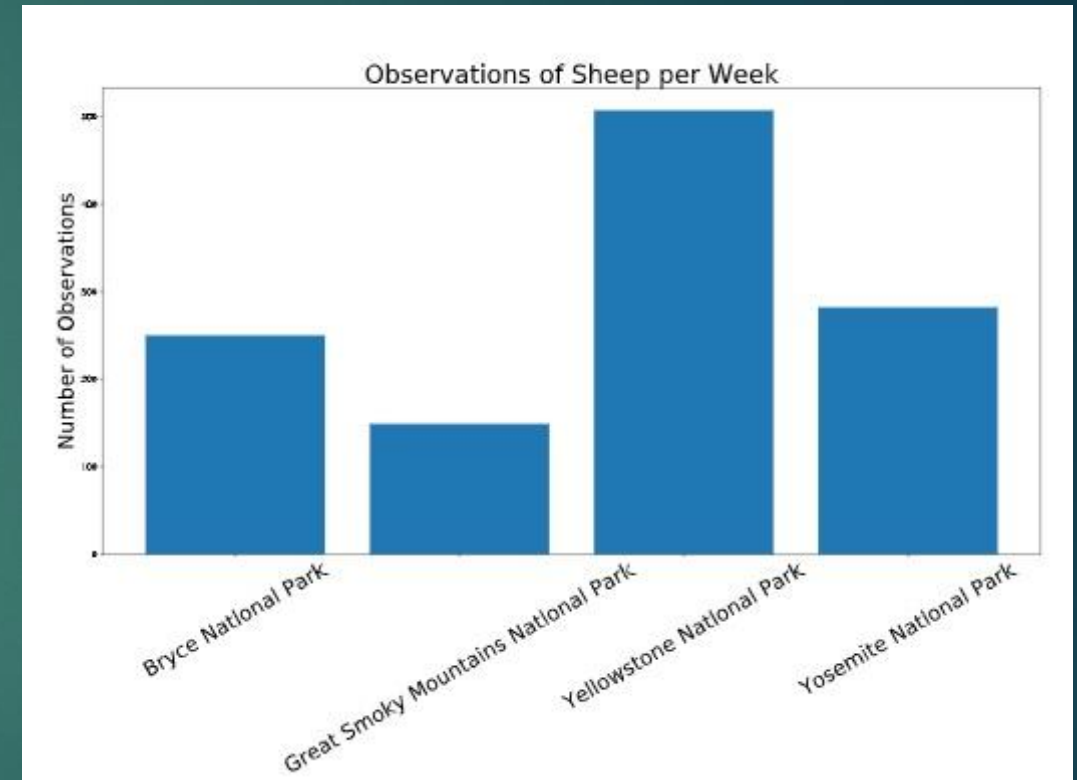
# Plotting sheep observations

```
plt.figure(figsize = (20,10))
ax = plt.subplot()
plt.bar(range(len(obs_by_park)) , obs_by_park.observations, fontsize = 25)
ax.set_xticks(range(len(obs_by_park)))
ax.set_xticklabels(obs_by_park.park_name , fontsize = 25 , rotation = 30)
plt.yticks(range(400000, 1500000,200000),fontsize = 40)
plt.ylabel('Number of Observations',fontsize=40)

plt.title('Observations of Sheep per Week', fontsize=30)
plt.show()
```

Notes: Since Yellowstone National Park has so many more sightings than the other parks, it can be assumed it will be much quicker to obtain the desired sample size. The calculations on the following page will support this claim.

Figure 12:



# Sample size calculation

Scientists are interested in monitoring the rates of 'foot and mouth disease'. Given a baseline rate of 15% infected sheep at Bryce National Park and a sample size calculator, the object is to calculate the 'minimum detectable effect' if we want to detect 5 percentage point increase.

$$\text{Min Dect' Effect} = 100.0 * \frac{0.05}{0.15} = 33.33 \%$$

Then, setting a significance level of 90% and using the sample size calculator, we obtain a sample size of 870 sheep.

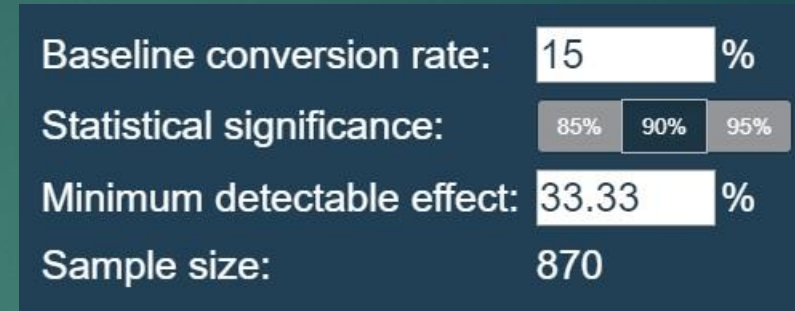
If there are 507 sheep sightings in Yellowstone park, how long would it take to obtain this calculated sample size?

$$\text{yellowstown\_weeks\_observing} = \frac{870 \text{ sheep}}{507 \text{ sheep/week}} = 1.72 \text{ weeks}$$

Repeat the same for Bryce National park.

$$\text{Bryce\_weeks\_observing} = \frac{870 \text{ sheep}}{507 \text{ sheep/week}} = 3.48 \text{ weeks}$$

Figure 13 : Sample size calculator



The image shows a sample size calculator interface with a dark blue background. It contains four rows of input fields and results:

- Baseline conversion rate:** A text input field containing '15' followed by a '%' symbol.
- Statistical significance:** Three radio button options: '85%', '90%' (which is selected), and '95%'.
- Minimum detectable effect:** A text input field containing '33.33' followed by a '%' symbol.
- Sample size:** A text input field containing '870'.