codecademy

Biodiversity of National Parks

Introduction to Data Analysis

TINA LUMI

21-10-2018

analyzing data

To get familiar with the data let's import necessary modules, load the data from csv file and inspect top 5 rows of our dataframe using ***.head()***

```
In [5]: from matplotlib import pyplot as plt
        import pandas as pd
```

```
In [6]: species = pd.read_csv('species_info.csv')
```

Output:

```
In [8]: species.head()
```

Out[8]:

	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), Dom...	NaN
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	NaN
4	Mammal	Cervus elaphus	Wapiti Or Elk	NaN

analyzing data

Finding out more about our data:

How many different species are in the `species` DataFrame?

```
In [9]: species.scientific_name.nunique()
```

```
Out[9]: 5541
```

What are the different values of `category` in `species`?

```
In [10]: species.category.unique()
```

```
Out[10]: array(['Mammal', 'Bird', 'Reptile', 'Amphibian', 'Fish', 'Vascular Plant',
       'Nonvascular Plant'], dtype=object)
```

What are the different values of `conservation_status`?

```
In [11]: species.conservation_status.unique()
```

```
Out[11]: array([nan, 'Species of Concern', 'Endangered', 'Threatened',
       'In Recovery'], dtype=object)
```

analyzing data

*To find out how many species have a certain status we can group our data by **conservation_status**.*
*Because empty cells are not being counted first set of results **(Results 1)** does not give us the full picture. To fix that we replace all empty cells in column **conservation_status** with '**No Intervention**' and group our data by **conservation_status** one more time which gives us second and accurate set of results (**Results 2**)*

```
In [12]: species.groupby('conservation_status').scientific_name.nunique().reset_index()

In [13]: species.fillna('No Intervention', inplace=True)

In [14]: species.groupby('conservation_status').scientific_name.nunique().reset_index()
```

Results 1:

	conservation_status	scientific_name
0	Endangered	15
1	In Recovery	4
2	Species of Concern	151
3	Threatened	10

Results 2:

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

analyzing data

*To create a visual representation of our findings let's sort data by **scientific_name** (count of species per **conservation_status**)*

In [17]:
```
protection_counts = species.groupby('conservation_status')\
    .scientific_name.count().reset_index()\
    .sort_values(by='scientific_name')

protection_counts.head()
```

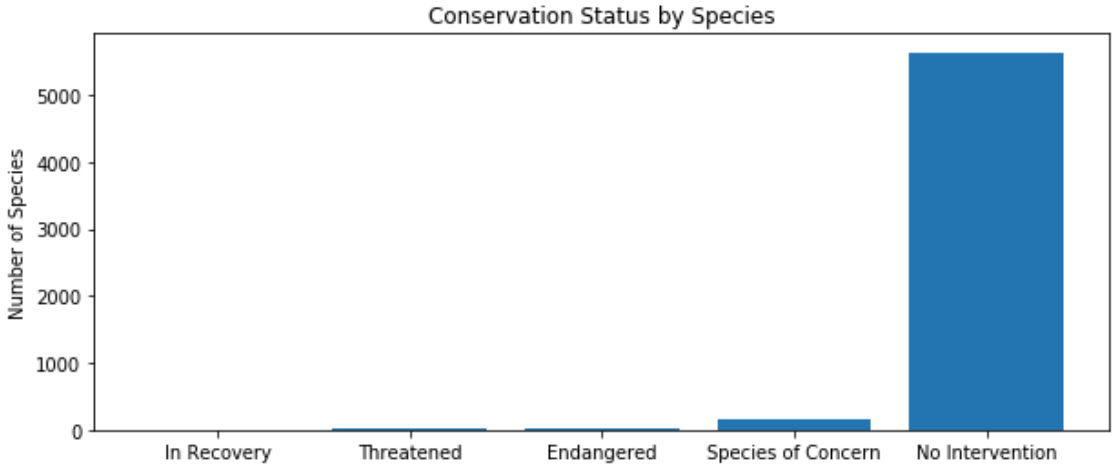
Out[17]:

	conservation_status	scientific_name
1	In Recovery	4
4	Threatened	10
0	Endangered	16
3	Species of Concern	161
2	No Intervention	5633

analyzing data

Graph of our data:

In [22]:
```
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)
plt.ylabel('Number of Species')
plt.title('Conservation Status by Species')
plt.show()
```



analyzing data

To find out which species are more likely to be endangered let's perform following modifications to our data:

- *create new column **is_protected** that is **True** if **conservation_status** is not equal to '**No Intevention**'*
- *group results by both **category** and **is_protected***
- *pivot table and rename columns*
- *add a column **percent_protected** that shows % of protected species*

In [24]:
```
species['is_protected'] = species.conservation_status != 'No Intervention'
```

In [28]:
```
category_counts = species.groupby(['category', 'is_protected']).scientific_name.nunique().reset_index()
```

In [32]:
```
category_pivot = category_counts.pivot(columns = 'is_protected',
                                       index = 'category',
                                       values = 'scientific_name')\
                                .reset_index()
```

In [34]:
```
category_pivot.columns = ['category', 'not_protected', 'is_protected']
```

In [36]:
```
category_pivot['percent_protected'] = category_pivot.is_protected /\
                                      (category_pivot.is_protected + category_pivot.not_protected)
```

analyzing data

Output:

```
In [37]: category_pivot
```

Out[37]:

	category	not_protected	is_protected	percent_protected
0	Amphibian	72	7	0.088608
1	Bird	413	75	0.153689
2	Fish	115	11	0.087302
3	Mammal	146	30	0.170455
4	Nonvascular Plant	328	5	0.015015
5	Reptile	73	5	0.064103
6	Vascular Plant	4216	46	0.010793

significance test

To find out if one set of data is significantly different from the other we need to run a significance test.
*Let's compare **Mammals to Birds** and **Mammals to Reptiles** and try to identify if one species is significantly more likely to be endangered than the other. In order to do so we need to perform Chi Square Test which requires taking following steps:*

- *importing relevant function from scypy*
- *creating contingency tables (**contingency** for Mammals to Birds comparison and **reptile_mammal** for Mammals to Reptiles comparison)*
- *analyzing p-values*

In [42]:
```
from scipy.stats import chi2_contingency
```

In [40]:
```
contingency = [[30, 146],
               [75, 413]]
```

In [43]:
```
chi2, pval, _, _ = chi2_contingency(contingency)
pval
```

Out[43]:
```
0.6875948096661336
```

In [45]:
```
reptile_mammal = [[5, 73],
                  [30, 146]]
chi2, pval, _, _ = chi2_contingency(reptile_mammal)
pval
```

Out[45]:
```
0.03835559022969898
```

significance test results

When performing Chi Square Test we take a null hypothesis that one dataset is not significantly different from the other and depending on the p-value we either reject or confirm it.

Comparing Mammals to Birds returned p-value > 0.05, which means that there is no significant difference between Mammals and Birds in terms of endangerment.

Comparing Mammals to Reptiles returned p-value < 0.05, which means there is a significant difference and Reptiles are significantly more endangered than Mammals.

```
In [42]: from scipy.stats import chi2_contingency

In [40]: contingency = [[30, 146],
                        [75, 413]]

In [43]: chi2, pval, _, _ = chi2_contingency(contingency)
         pval

Out[43]: 0.6875948096661336

In [45]: reptile_mammal = [[5, 73],
                           [30, 146]]
         chi2, pval, _, _ = chi2_contingency(reptile_mammal)
         pval

Out[45]: 0.03835559022969898
```

analyzing observations

We received a new set of data that provides us with recorded observations of different species over the period of 7 days. Let's familiarize ourselves with this data:

```
In [47]: observations = pd.read_csv('observations.csv')
         observations.head()
```

Out[47]:

	scientific_name	park_name	observations
0	Vicia benghalensis	Great Smoky Mountains National Park	68
1	Neovison vison	Great Smoky Mountains National Park	77
2	Prunus subcordata	Yosemite National Park	138
3	Abutilon theophrasti	Bryce National Park	84
4	Githopsis specularioides	Great Smoky Mountains National Park	85

analyzing observations

To find out how many times sheep have been spotted in a specific park during the last 7 days we need to make following modifications to our data:

- *create a new column **is_sheep** in table **species** which will be **True** if it is a sheep (look for '**sheep**' in column **common_name**)*
- *create new table **sheep_species** that will only include **Mammals** with '**sheep**' in their **common_name***
- *merge tables **sheep_species** and **observations** to create a dataframe to work with*
- *group new table by **park_name** and show cumulative number of observations per park per species*
- *visualize data*

```
In [61]: species['is_sheep'] = species.apply(lambda row: 'sheep' in row['common_names'].lower(), axis=1)
         species.head()
```

```
In [66]: sheep_species = species[(species.is_sheep == True) & (species.category == 'Mammal')]
         sheep_species
```

```
In [67]: sheep_observations = observations.merge(sheep_species)
         sheep_observations
```

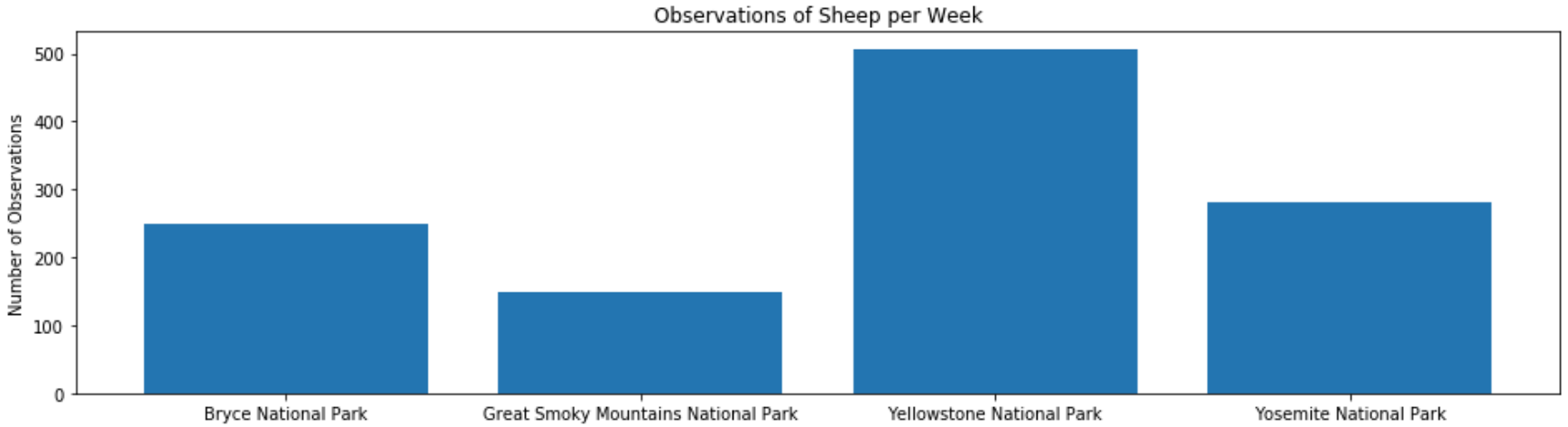
```
In [68]: obs_by_park = sheep_observations.groupby('park_name').observations.sum().reset_index()
         obs_by_park
```

analyzing observations

Output:

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282

```
plt.figure(figsize=(16, 4))
ax = plt.subplot()
plt.bar(range(len(obs_by_park.park_name)), obs_by_park.observations)
ax.set_xticks(range(len(obs_by_park.park_name)))
ax.set_xticklabels(obs_by_park.park_name)
plt.ylabel('Number of Observations')
plt.title('Observations of Sheep per Week')
plt.show()
```



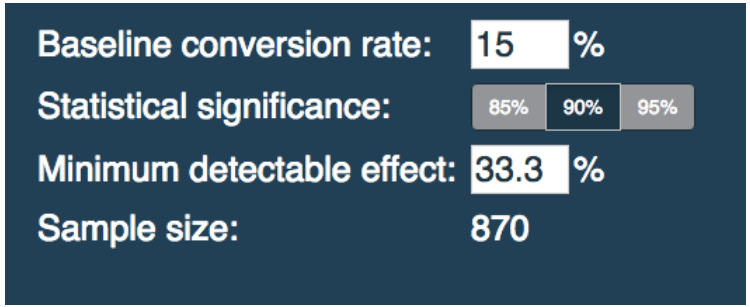
sample size determination

To determine sample size we need to do the following:

- *calculate Minimum Detectable Effect*
- *find sample size using sample size calculator*

```
In [83]: min_detectable_effect = 5. / 15 * 100
         min_detectable_effect

Out[83]: 33.33333333333333
```



Minimum Detectable Effect = 33,33%
Sample Size = 870

sample size determination

To define how many weeks scientists need to observe sheep in Bryce National Park and Yellowstone National Park we do following calculations:

In [107]:
```
#baseline conversion rate = 15%
#statistical significance = 90%
#minimum detectable effect = 33.33%
#sample size = 870

sample_size = 870

obs_bryce_park = obs_by_park[(obs_by_park.park_name == 'Bryce National Park')].observations
bryce_park = sample_size / float(obs_bryce_park)
bryce_park
```

Out[107]:
```
3.48
```

In [108]:
```
obs_yellowstn_park = obs_by_park[(obs_by_park.park_name == 'Yellowstone National Park')].observations
yellowstn_park = sample_size / float(obs_yellowstn_park)
yellowstn_park
```

Out[108]:
```
1.7159763313609468
```

Scientists will need to observe sheep for 3,5 weeks (24 days) in Bryce National Park and for 1,7 weeks (12 days) in Yellowstone National Park.