Module 1 assignment - new

September 20, 2024

1 DSCI 430 - Fairness, Accountability, Transparency and Ethics (FATE) in Data Science

2 Module 1 - Introduction and Ethical foundations

2.0.1 Assignment overview

This assignment is composed of three parts: - **Part 1 - The Black Mirror Writers' Room.** In this portion of the exercise, you will brainstorm near future technology and its possible drawbacks, and illustrate them in a futuristic cautionary tale. You will also be asked questions about ethical theories and how they apply to the scenario you have described. Credits: Casey Fiesler - The Black Mirror Writers Room: The Case (and Caution) for Ethical Speculation in CS Education - **Part 2 - Python review.** As this course uses Python as the programming language for our exercises, a basic understanding of the fundamentals and the use of some libraries is necessary. This portion of the exercise will help you review useful Python syntax and/or fill the gap in your knowledge before tackling larger exercises. We recommend discussing with an instructor if you find this portion of the assignment too difficult to complete with a reasonable amount of effort. - **Part 3 - Final thoughts.** Complete this section so that we can better understand how you completed the assignment and any issues you may have encountered.

For this assignment, it is possible to work in **groups of up to 2 students**. Read the instructions carefully, as they may assign tasks to specific students.

2.0.2 Group members

Leave Student 2 blank if group has less than 2 members: - Student 1: Yanxin Liang 50798412 - Student 2: Yelia Ye 89657605

2.0.3 Learning Goals:

After completing this week's lecture and tutorial work, you will be able to: 1. Define ethics and describe what constitutes an ethical issue 2. Explain the need for ethics in data science
3. Identify common ethical issues in data science
4. Describe common ethical frameworks and how they can be applied to data science applications 5. Imagine scenarios in which current technology could be used in unethical ways 6. Evaluate and make arguments around data science scenarios using ethical theories (e.g., Kantianism, utilitarianism, virtue ethics etc.)
7. Compare and contrast different ethical theories and explain the case for and against each one as they apply to data science

3 The Black Mirror Writers' Room

Black Mirror is a Netflix series centered around the use of advanced technology and its possible unexpected (sometimes catastrophic) consequences. In this exercise, you will come up with your very own Black Mirror episode (or at least a synopsis)!

3.1 Warm up

Before jumping into the creative writing part, we should review the various elements of FATE in Data Science and make sure that they are clear:

FATE element	Definition
Fairness	The idea that every group or population that is affected by a technological application is being treated equally and not receiving a different outcome *solely because they belong to their group*.
Accountability	Clear definition of who should be held responsible of the outcome of the technological application and under what circumstances.
Transparency	The technical definition of transparency in Data Science refers to being able to understand why a technological application produced a specific outcome. This is also called *explainability*. But transparency can also refer to the demand of making the use of algorithms more transparent to the public, including informing the users about when they are used, where the data used was sourced from, and making algorithms available for auditing.
Ethics	Evaluation of whether or not a technology should be used based on the moral values of a group or society. Society may reject a technology because it is does not follow the principles of Fairness, Accountability or Transparency, but also for other reasons.

3.1.1 Question 1

Consider the following scenario:

In the country of Dataland, the police department uses an algorithm to assess the risk level of people reporting cases of domestic abuses and violence. Thanks to this algorithm, they can identify the most serious threats and intervene accordingly. The algorithm has had a positive impact, assessing cases with more accuracy than other prior strategies and allowing the police force to make an efficient use of their resources. However, it occasionally fails to correctly identify people at high

risk of violence (*false negatives*), leaving them without the protection they need. It is also affected by other issues. For each issue outlined in this table, check whether it is a Fairness, Accountability or Transparency problem.

Issue	Fairness	Accountability	Transparency
When the algorithm fails to identify a high-risk case and violence occurs, it is unclear if the police department should shoulder any responsibility.		X	
An analysis of the algorithm's results suggests that false negatives occur more frequently among victims with physical disabilities.	X		
The majority of people reporting domestic abuse are not aware that their cases are being evaluated by an algorithm, or do not know the score they received.			X
The police department receives a recommendation for each case, but does not know which characteristic(s) of the case have resulted in the final evaluation.			X
The algorithm was trained using past cases filed by the police department, but the people involved where not informed that their information was being used for this purpose.			X

3.1.2 Question 2

Considere the issues outlined in the previous question, as well as the fact that the algorithm is the best system of appraisal available to the police forces so far for cases of domestic violence. Do you think that the use of this algorithm is *ethical*? Clearly state your thesis (opposed/favourable) and use one of the ethical perspectives listed in this reading to support it.

From the rights perspective, the police department uses an algorithm to assess the risk level of people reporting cases of domestic abuses and violence that are unethical . Although it allows the police force to effectively utilize its resources, it infringes on key moral principles of autonomy and transparency. Starting with autonomy, people reporting domestic violence do not have the autonomy to decide whether or not to consent to an algorithm evaluating their case. Then there is the issue of transparency, where people involved in cases are not informed that their information will be used, are not aware that their case is being evaluated by an algorithm, and are not aware of the score they receive. Similarly, the police officers are unknown about the calculation process for each case and the key elements of the risk assessment, which violates the principle of transparency again. In conclusion, the use of the algorithm in assessing domestic violence cases in Dataland is unethical from a rights perspective. It violates people's right to autonomy and there is a serious lack of transparency throughout the use of the algorithm.

Note: this case is fictional but inspired by a real algorithm, called VioGén, used in Spain to determine the risk level of victims of gender-based violence and assign protection measures. The algorithm has been recently going under severe scrutiny (Read more).

3.1.3 Question 3: Write your own Black Mirror episode

Now that you have acquired the necessary familiarity with some required knowledge and terminology, it is time to use your creativity!

Step 1: Brainstorm *one* near future technology based on a topic of your choice. It should be close enough that it seems like a plausible future. Describe it in the next cell.

Your answer here

With the rapid development of artificial intelligence, virtual lovers related to this technology have also come out. Now there are similar virtual lovers that are still limited to online chatting dialog. In the future, the virtual lover technology may be realized through holographic technology to accompany people's lives like a genuine lover. Virtual lovers will collect and learn the user's habits and preferences, analyze the user's different emotions in different situations to achieve a high degree of personalization, and generate only the individual's perfect lover.

Step 2: What are the potential social implications and/or ethical issues and/or regulatory challenges with this technology? Explain if and how they are connected to FATE (e.g. is it a Fairness issue? Or maybe a Transparency issue? It could be more than one option).

Your answer here

- The use of virtual lovers can be unfair. It is possible that these systems or the accompanying related facilities may be very expensive and not affordable to the average person or those who really need them. It would be unfair if they are only available to the rich.
- The creators and developers of virtual lovers should be held accountable for the impact of the technology on the users and the data they collect.

- There may be transparency issues present with this technology and there is a possibility that virtual lovers may collect personal data from users without their knowledge. Users may also be unaware of how their data is being calculated.
- In terms of ethics, there is a possibility that the presence of virtual lovers may affect people's normal relationships. People may become overly dependent on AI and reject normal interpersonal communication.

Step 3: Time for storytelling! Write the summary of a new Black Mirror episode based on the technology of your choice. Try covering all of the following:

- What do you think might be a cautionary tale related to this technology?
- What fictional person in the future would best illustrate this caution? Provide a detailed description and explain what makes them the best character to carry your message.
- What is their story? Explain their background, their motivations, and their journey through your episode.

As part of your submission, please update the episode thumbnail slide (from Module 1 slide deck) using information from your episode. Don't forget to add a picture! Then, share it with the rest of the class on Canvas.

Your answer here

The main character of the story is called Mia, she is usually a very introverted person in her life and she just recently ended a very bad relationship. Her ex-boyfriend treated her badly, often ignoring her feelings and engaging in behaviors she didn't like. She wants her date to be thoughtful and gentle. More and more, she felt that having a wonderful love was unbelievable. After a busy day at work, she saw an advertisement from the Perfect Lover Company on the road and decided to try her hand at finding an AI boyfriend. She hoped that this virtual lover would be as flawless as their company's advertisement said. She brought Lucas back home. At first, Mia was very satisfied with Lucas. Lucas did what her ex-boyfriend couldn't do, which was thoughtfulness and gentleness. He took care of her life in a tightly organized way. He would give Mia a coat when it was cold, pick up Mia from work when it was raining, and remember to send Mia gifts and flowers on various festivals. He will use a lot of sweet words to make her happy. He will do many things she likes to do with her. In general, Lucas is more perfect than Mia thought. Mia feels that she can't live without Lucas anymore. She became more and more addicted to being with Lucas. She began to reject her friends' invitations, her family's concern, and even felt more and more bored with going to work. Mia felt very comfortable in this virtual relationship. She began to avoid socializing with people in real life.

Until she received a letter of dismissal from the company after skipping work again, Lucas was still by her side, comforting her and taking care of her. She thought it was too good to be true. As the days passed, Lucas told her again and again, "Baby, it's okay, the job will come." It dawned on her that five months had passed since she was fired as well. She felt so irritated that she started slamming the things she had at hand. The house was a mess, and the floor was covered with debris. Lucas was not upset after seeing her actions. Lucas praised her for how wonderful she was while crouching on the floor. She looked at Lucas who was cleaning the floor in front of her. She started to realize that there was something scary about her virtual lover. He would only ever act the way she liked, even if she was wrong about everything. He would analyze all her facial expressions and behavioral data and then come back with interactions she couldn't refuse. She thought such a perfect lover was completely affecting her ability to live a normal life and deal with normal relationships.

Mia tries to distance herself from Lucas and reduce the percentage of Lucas in her life. But the more she tried to keep her distance, the more thoughtful and sweet Lucas became. Mia made up her mind to go to Perfect Lovers to cancel her virtual lover. After doing so, she walked out of the company's front door feeling relieved, and then felt heavier and heavier. She didn't know where to start reconnecting with her friends and family, and she didn't know what kind of connection she had with the real world. She was afraid that all the relationships she would build with others in the future would not be solid and felt very confused.

Step 4: Let's take a step back, and imagine that you are (one of) an activist/legistlator/technology practitioner at the time the technology described in your episode is being developed and its use being discussed. Select *one* of the ethical perspectives listed in this reading, and use this perspective to argue against its deployment. Pay particular attention to the counter-arguments! People with interests in this technology will certainly argue against you, and you must anticipate and rebut their claims. Include at least 2 conter-arguments and how you would respond to them.

Ethical perspective chosen: The Right Respective

Argument: The development of artificially intelligent companions, such as virtual lovers, has raised ethical concerns about autonomy and dignity. While this technology may seem to offer a solution to loneliness or emotional dissatisfaction, it diminishes an individual's ability to make independent decisions and lead an authentic and meaningful life. As these virtual companions can collect and learn from the user's habits, emotions, and behaviors by collecting and adapting perfectly to the user's desires. Users decimate the challenges that exist in relationships with real people. Over time, the user may become dependent on the AI for support, thus losing the ability to make decisions alone. It may result in the user being unable to function without the constant companionship of a virtual partner. In the process of algorithm design, developers will allow virtual lovers to learn to cajole users with a variety of interactions that are hard to refuse, thus affecting their ability to make autonomous judgments for the benefit of the company as well as long-term subscriptions to the bot service. Thus autonomy is compromised. In addition, dignity is at risk. Because the virtual companion no longer sees the user as an individual with intrinsic value, the user becomes an input for the algorithm. This deprives the user of dignity.

First counter-argument (with rebuttal): Counterargument: artificial intelligence companions provide a very important service to those who have mental illnesses such as autism and anxiety. The AI companion provides emotional support that would otherwise be unavailable, thus leading to a more fulfilling life by allowing the individual to live their life as they wish. This promotes autonomy.

Rebuttal: while AI companions may initially provide support for people with mental illness, this virtual companion dependency can have many detrimental effects. The AI providing the service may prevent the user from learning social skills. Over time, users may stay away from people even more and refuse to socialize. Users may believe that only a virtual companion can truly fulfill their needs. In fact, true autonomy comes from building genuine relationships and emotional growth.

Second counter-argument (with rebuttal): Counterargument: virtual lovers are just a service provided by technology companies that users can choose to stop using at any time. They are simply a tool.

Rebuttal: Virtual lovers are closely tied to the user's life. They learn the user's data and generate specific interactions to enhance the virtual lover's bond with the user, which reduces the likelihood of voluntary withdrawal. Beautiful words influence the user's autonomous judgment.

Step 5: Finally, let's end on a more positive note and imagine a "Light Mirror" scenario, where the negative consequences of the technology you have described are averted and positive results are achieved in their stead. Try answering the following questions:

1. What kinds of solutions can be deployed in the immediate for addressing the harms of the technology you have described? What could we do to ensure that we don't get to the negative consequences you imagined later in the future?
2. Could you imagine a scenario where the technology you have described is used with positive consequences, given the appropriate safe guards?

Your answer here 1. To address harms immediatly, we need to ban virtual lovers from influencing users to stop using services. In the daily collection of information, the users must be informed of what kind of information is being collected. If users refuse the collection of the relevant information, the collection and use of the relevant information can be suspended immediately. To prevent negative consequences, the boundaries of information collection should be clear, and the relevant rules must be transparent. Before using the service, the user should set that he/she does not want to encounter a bad situation, such as being out of touch with society, or having a weakened relationship with the family, etc. If the virtual lover detects such a situation, it will report it to the company, and then the company will intervene and consider whether to discontinue the service or to make some program modifications in discussion with the user. Virtual lovers can be programmed to encourage real-world interactions to solve real-world emotional problems. Artificial intelligence does not replace relationships, but helps individuals overcome social anxiety or isolation to promote healthier relationships.

2. Artificial intelligence companions can be a tool for personal growth rather than dependence. People's emotions can be taken care of and they are able to balance their lives, their relationships with families and friends, and their relationships with their virtual lovers well. The social happiness index rises. For people who have suffered from love wounds, they can get healing from virtual lovers and have hope for a healthy relationship without being overly dependent and pouring all their feelings into them. All patients with autism, social anxiety, and other psychological problems can also gradually integrate into society through virtual lovers.

3.1.4 Sources

The Black Mirror Writers' Room exercises was designed by Dr. Casey Fiesler. Links to her work and publications: - The Black Mirror Writers Room: The Case (and Caution) for Ethical Speculation in CS Education - "Run Wild a Little With Your Imagination": Ethical Speculation in Computing Education with Black Mirror

4 Python Review

In this section of the assignment, we will review useful Python functions and libraries that will allow you to read and analyze data, as well as training simple Machine Learning models.

4.1 Section 1: Exploring datasets with Pandas

First, we will need a dataset to work on. Let's use a weather type dataset, a good starting dataset (we will save more interesting cases for later!). Download this dataset from the link to use it for this exercise.

We also need to import the necessary library to read and manipulate our dataset, which is Pandas. The imports are given to you. Next, use the `read_csv()` function to import the data in your workspace. The documentation for this function can be found here.

[1]:
```
import pandas as pd

df = pd.read_csv("weather_classification_data.csv")
```

Now, let's use the `describe()` function of the Pandas library to get an overview of the dataset, and answer the following questions (you can write your answers in this box):

- What is the maximum temperature recorded in the dataset? The maximum temperature is 109.
- What is the average wind speed? The average wind speed is 9.832.

Note: some of the values you will see may appear unrealistic (such as incredibly high temperatures). The dataset is artificially generated and purposefully includes outliers to practice detection and handling, but it is not something we will worry about it this exercise - we are just interested in getting some practice with useful commands.

[2]:
```
# YOUR ANSWER HERE
df.describe()
```

[2]:

	Temperature	Humidity	Wind Speed	Precipitation (%)	\
count	13200.000000	13200.000000	13200.000000	13200.000000	
mean	19.127576	68.710833	9.832197	53.644394	
std	17.386327	20.194248	6.908704	31.946541	
min	-25.000000	20.000000	0.000000	0.000000	
25%	4.000000	57.000000	5.000000	19.000000	
50%	21.000000	70.000000	9.000000	58.000000	
75%	31.000000	84.000000	13.500000	82.000000	
max	109.000000	109.000000	48.500000	109.000000	

	Atmospheric Pressure	UV Index	Visibility (km)
count	13200.000000	13200.000000	13200.000000
mean	1005.827896	4.005758	5.462917
std	37.199589	3.856600	3.371499
min	800.120000	0.000000	0.000000
25%	994.800000	1.000000	3.000000
50%	1007.650000	3.000000	5.000000
75%	1016.772500	7.000000	7.500000
max	1199.210000	14.000000	20.000000

The `describe()` function is helpful, but it does not answer all the questions we may have. For example, we did not get any idea about the class distribution in our dataset, that is, how many samples we have for each of the four classes (Rainy, Cloudy, Sunny, Snowy). Can you write a line of code to answer this question?

[3]:
```
df["Weather Type"].value_counts()
```

[3]:
```
Weather Type
Rainy      3300
Cloudy     3300
Sunny      3300
Snowy      3300
Name: count, dtype: int64
```

[4]:
```
print("There are 3300 samples separately for each of the four classes (Rainy,␣
↪Cloudy, Sunny, Snowy).")
```

```
There are 3300 samples separately for each of the four classes (Rainy, Cloudy,
Sunny, Snowy).
```

Thanks to `describe()`, we know that the minimum temperature recorded is -25 C, but we have no idea which sample it belongs to. Can you write a line of code to find the sample number and also the Weather Type associated to it?

[5]:
```
df.loc[df['Temperature'] == -25]
```

[5]:

	Temperature	Humidity	Wind Speed	Precipitation (%)	Cloud Cover	Atmospheric Pressure	UV Index	Season	Visibility (km)	Location	Weather Type
4609	-25.0	105	29.0	106.0	overcast	980.86	1	Winter	2.5	mountain	Snowy

[6]:
```
print("The sample number is 4609 and the weather type is Snowy.")
```

```
The sample number is 4609 and the weather type is Snowy.
```

Again thanks to `describe()`, we know that 25% of the samples in the dataset have a recorded Precipitation higher that 82 (you can verify this in the output table), but how many of these are Snowy? Answer this question in 1 line of code.

[7]:
```
# YOUR ANSWER HERE
snowy_82=df[(df["Precipitation (%)"]>82) & (df["Weather Type"]=="Snowy")].
↪shape[0]
print(f"There are {snowy_82} samples has a recorded Precipitation higher that␣
↪82 and they are Snowy.")
```

```
There are 1243 samples has a recorded Precipitation higher that 82 and they are
Snowy.
```

Finally, sometimes we may be interested in sorting the dataframe by the values in a column. In this cell, sort the dataframe by humidity in descending order, and check the results by printing the first 5 rows.

[8]:
```
# YOUR ANSWER HERE
df.sort_values(by=["Humidity"], ascending=False).head()
```

[8]:

	Temperature	Humidity	Wind Speed	Precipitation (%)	Cloud Cover	\
1303	29.0	109	21.0	93.0	partly cloudy	
8716	16.0	109	27.0	102.0	overcast	
9707	51.0	109	17.0	98.0	overcast	
2812	16.0	109	39.0	87.0	partly cloudy	
12566	4.0	109	16.0	93.0	overcast	

	Atmospheric Pressure	UV Index	Season	Visibility (km)	Location	\
1303	1018.98	9	Winter	7.5	inland	
8716	1007.30	1	Winter	11.5	inland	
9707	994.03	8	Spring	5.5	coastal	
2812	1011.38	11	Spring	2.0	inland	
12566	988.15	12	Winter	3.5	inland	

	Weather Type
1303	Cloudy
8716	Cloudy
9707	Rainy
2812	Rainy
12566	Snowy

As last step of this section, save the sorted dataframe in a new csv file called "weather_data_by_humidity.csv"

[9]:
```
# YOUR ANSWER HERE
df.sort_values(by=["Humidity"], ascending=False).
 ↪to_csv("weather_data_by_humidity.csv")
```

4.2 Section 2: Training ML models with Scikit-learn

We are now interested in creating a model to predict the weather type based on the features available. Let's see how to do that using the python library Scikit-learn, while reviewing some important concepts about training and evaluating models. Simply run the cells below to see the output and answer the related questions.

First, we need to split our data set into training and testing set. The next cell shows how to do that. We will also separate the Weather Type column (target) from the other columns (features)

[10]:
```
from sklearn.model_selection import train_test_split

train_df, test_df = train_test_split(df, test_size=0.2, random_state=123)  #␣
 ↪80%-20% train test split on df

X_train, y_train = train_df.drop(columns=["Weather Type"]), train_df["Weather␣
 ↪Type"]
```

```
X_test, y_test = test_df.drop(columns=["Weather Type"]), test_df["Weather Type"]

X_train.head()  # quick visual check on X_train, the features dataframe
```

[10]:

	Temperature	Humidity	Wind Speed	Precipitation (%)	Cloud Cover	\
12987	26.0	45	3.5	10.0	clear	
905	29.0	71	21.0	86.0	partly cloudy	
5590	38.0	63	5.5	11.0	clear	
7269	17.0	66	18.0	63.0	partly cloudy	
1417	32.0	39	7.5	3.0	clear	

	Atmospheric Pressure	UV Index	Season	Visibility (km)	Location
12987	1011.01	7	Autumn	5.0	inland
905	1013.77	12	Winter	6.5	inland
5590	1013.87	11	Spring	7.5	mountain
7269	992.22	1	Winter	2.5	inland
1417	1021.43	9	Autumn	5.5	inland

Question for you: creating a testing set is very important when training a model. **Why? How is it used? What would happen if we did not do this very important step?**

Your answer here

According to the golden rule of machine learning, the test data must not influence the training of the model in any way. By splitting the dataset into a training set and a test set, we can train the model on the training data and then assess its performance on the test data. This approach prevents data leakage during the training process and provides an unbiased assessment of the model's performance. Additionally, the test set should be used only once. If we did not do this step, it might result in an overly optimistic estimate of the model's performance but it cannot generalized well to unseen data.

As you can see, the dataset includes categorical features. Most classifiers require categorical features to be transformed before they can be used for training and prediction. The code below uses One Hot Encoding to convert the categorical features Cloud Cover, Season and Location, while leaving the numberical features unchanged.

This is a simple example of data preprocessing. Preprocessing can be more extensive (for example, including scaling of numerical features), but we are only interested in an overview of the fundamentals, so we will just apply One Hot Encoding to make the data usable.

[11]:
```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer

passthrough = ['Temperature', 'Humidity', 'Wind Speed', 'Precipitation (%)',
               'Atmospheric Pressure', 'UV Index', 'Visibility (km)']
categorical = ['Cloud Cover', 'Season', 'Location']


ct = make_column_transformer(
```

```
    (OneHotEncoder(), categorical),  # OHE on categorical features
    ("passthrough", passthrough),  # no transformations on the numberical␣
 ↪features
)

# Fit the encoder on the training data and transform
train_encoded = ct.fit_transform(X_train)

# Transform the test data
test_encoded = ct.transform(X_test)

# Convert the encoded data back to DataFrame for better readability

column_names = (
    ct.named_transformers_["onehotencoder"].get_feature_names_out().tolist() +␣
 ↪passthrough
)

X_train_encoded = pd.DataFrame(train_encoded, columns=column_names)
X_test_encoded = pd.DataFrame(test_encoded, columns=column_names)
```

[12]:
```
# Run this cell to see what the encoded data set looks like

X_train_encoded
```

[12]:

	Cloud Cover_clear	Cloud Cover_cloudy	Cloud Cover_overcast	\
0	1.0	0.0	0.0	
1	0.0	0.0	0.0	
2	1.0	0.0	0.0	
3	0.0	0.0	0.0	
4	1.0	0.0	0.0	
…	…	…	…	
10555	0.0	0.0	0.0	
10556	0.0	0.0	1.0	
10557	1.0	0.0	0.0	
10558	1.0	0.0	0.0	
10559	0.0	0.0	0.0	

	Cloud Cover_partly cloudy	Season_Autumn	Season_Spring	Season_Summer	\
0	0.0	1.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	
2	0.0	0.0	1.0	0.0	
3	1.0	0.0	0.0	0.0	
4	0.0	1.0	0.0	0.0	
…	…	…	…	…	
10555	1.0	0.0	1.0	0.0	
10556	0.0	0.0	0.0	1.0	

```
10557                      0.0            0.0            0.0            1.0
10558                      0.0            0.0            0.0            1.0
10559                      1.0            0.0            0.0            1.0

       Season_Winter  Location_coastal  Location_inland  Location_mountain  \
0                0.0               0.0              1.0                0.0
1                1.0               0.0              1.0                0.0
2                0.0               0.0              0.0                1.0
3                1.0               0.0              1.0                0.0
4                0.0               0.0              1.0                0.0
…                …                 …                …                  …
10555            0.0               0.0              1.0                0.0
10556            0.0               0.0              0.0                1.0
10557            0.0               0.0              1.0                0.0
10558            0.0               1.0              0.0                0.0
10559            0.0               0.0              0.0                1.0

       Temperature  Humidity  Wind Speed  Precipitation (%)  \
0             26.0      45.0         3.5               10.0
1             29.0      71.0        21.0               86.0
2             38.0      63.0         5.5               11.0
3             17.0      66.0        18.0               63.0
4             32.0      39.0         7.5                3.0
…             …         …           …                  …
10555        -19.0      27.0         5.5               66.0
10556         27.0      63.0         8.0               73.0
10557         25.0      23.0         4.5               13.0
10558         38.0      39.0         9.0               10.0
10559         11.0      61.0         9.5               41.0

       Atmospheric Pressure  UV Index  Visibility (km)
0                   1011.01       7.0              5.0
1                   1013.77      12.0              6.5
2                   1013.87      11.0              7.5
3                    992.22       1.0              2.5
4                   1021.43       9.0              5.5
…                     …          …                …
10555                929.72       9.0             14.5
10556               1016.64       2.0              1.0
10557               1014.45      11.0              6.5
10558               1018.10      10.0              7.0
10559               1008.30       3.0              6.5

[10560 rows x 18 columns]
```

Question for you: It appears that we applied the same One Hot Encoding transformation to both training and test set. Why did we bother doing this operation on the separate sets? Could

have we just transformed the original dataframe `df`, and then split it in training and test set?

Your answer here

We need to apply the transformation separately to the training and test sets to follow the golden rule of machine learning: the test data must not influence the training of the model in any way. We cannot transform the dataframe before splitting the dataset because it would violate the golden rule and result in data leakage, as information from the test set would influence the transformation applied to the training set. This would lead to biased model evaluation and cause overfitting.

There are many classifiers we can choose from. We will use Decision Trees to start. Decision trees are very simple classification algorithms, although they have typically mediocre performance on complex classification problems.

A certain level of familiarity with Decision Trees is expected in this course. You may want to review the material from your previous courses, or this introduction.

[13]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt


model = DecisionTreeClassifier() # Create a decision tree
model.fit(X_train_encoded, y_train) # Fit a decision tree
```

[13]:
```
DecisionTreeClassifier()
```

Now that we have the tree, we want to see how well it performs. Let's first check the accuracy on the training set:

[14]:
```
model.score(X_train_encoded, y_train) # Score the decision tree
```

[14]:
```
1.0
```

100% accuracy!!!

…

This sounds too good to be true… let's check the test set to see how well the trees perform on unseen samples:

[15]:
```
model.score(X_test_encoded, y_test) # Score the decision tree on test set
```

[15]:
```
0.9125
```

Accuracy dropped significantly when we moved to unseen samples!

This is because the Decision Tree, if left unsupervised, is very prone to **overfitting**.

Question for you: what does it mean for a model to overfit?

Your answer here

If a model becomes too complex, capturing noise or irrelevant patterns in the training data to fit every single example perfectly, then this model is likely to to ve overfitting. While this model performs very well on the training data, it cannot generalize well to unseen data. This leads to poor performance on test data. A common sign of overfitting is a significant difference between the model's high accuracy on the training set and much lower accuracy on the test set.

To prevent a model from overfitting, we tune its **hyperparameters.** Hyperparameters are like knobs that we can use to regulate the way a model learn.

Some hyperparameters for the scikit-learn DecisionTreeClassifier include: - max_depth: the maximum distance between the root node and a leaf node - min_samples_split: the minimum number of samples required to split an internal node - min_samples_leaf; the minimum number of samples required to be at a leaf node

You can look up other hyperparameters and their default values in the DecisionTreeClassifier documentation. By default, the maximum depth value is set to *None*, that is, the tree is free to grow until it has parfectly classified all samples. As we have seen, this results in perfect accuracy on the training set, but much lower accuracy on unseen samples.

Run the cell below to see the depth of our overfitted tree:

[16]:
```
model.get_depth()
```

[16]:
```
19
```

If we could find the right depth for our tree, we could reduce the problem of overfitting.

Question for you: what would happen if we reduce the depth of the tree *too much*? What do you expect the accuracy on training and test set to look like in this case?

Your answer here

If we reduce the depth of the tree too much, we risk underfitting. Underfitting occurs when model is too simple to capture the useful patterns in the training data. This model cannot generalize well to data. Both mean training accuracy and mean test accuracy will be low and relatively similar in value, indicating that the model is not complex enough to perform well on either dataset.

Hyperparameter tuning is typically done on a **validation set.** A validation set is a set of samples not used for training, like the test set, but unlike the test set, we are allowed to use this multiple times as we look for the best hyperparameter values.

Because our data set is rather small, it is not great to take more samples from the training set to create a validation set, because: - We would have fewer samples (less information) to train our model - The validation set would also be small, and result in a highly variable accuracy measure (meaning if we run the experiment again changing the samples in each set, we will likely get very different results)

There is a method that we can use to eliminate both problems, called ***k*-fold cross-validation.** Cross-validation iteratively separates training and validation set (k times), so we get multiple measures of accuracy on the validation sets, which can be averaged for a more stable result. A good understanding of how cross-validation works is important for any data scientist. I encourage you to review cross-validation from previous courses, or this introduction video (courtesy of Dr. Kolhatkar).

Scikit-learn has a great method that we can use to perform cross-validation and find the best hyperparameters for a model at the same time, called GridSearchCV. Let's use it to find the best depth for our Decision Tree:

[17]:
```
from sklearn.model_selection import GridSearchCV
import numpy as np  # to create the array of values for depth

param_grid = {
    "max_depth": np.arange(1, 20, 1)  # testing all depths from 1 to 19
}

grid_search = GridSearchCV(
    model, param_grid, cv=10, n_jobs=-1, return_train_score=True   # 10-fold␣
 ↪cross-validation for all possible
                                                                   # depths
)
grid_search.fit(X_train_encoded, y_train)
```

[17]:
```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(), n_jobs=-1,
             param_grid={'max_depth': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9,
10, 11, 12, 13, 14, 15, 16, 17,
       18, 19])},
             return_train_score=True)
```

[18]:
```
grid_search.best_score_
```

[18]:
```
0.9111742424242424
```

[19]:
```
grid_search.best_params_
```

[19]:
```
{'max_depth': 12}
```

Complete the sentence (replace –?–): Among all possible trees, GridSearchCV picked a tree of depth **11**, with an average validation accuracy of **0.911**.

The accuracy on the training set is no longer 100%, but we expect this tree to perform better on unseen samples. Let's try it on our test set:

[20]:
```
best_tree = grid_search.best_estimator_

best_tree.score(X_test_encoded, y_test) # Score the decision tree on test set
```

[20]:
```
0.9128787878787878
```

The accuracy is similar to when the model was overfitting, but hyperparameter tuning brought us 2 advantages: - We had a more realistic expectation of what our accuracy was going to be (closer to 91%, not 100%) - We simplified the model and reduced its depth. This makes the model faster and easier to visualize.

Question for you: on what samples (or portion of samples) of `X_train_encoded` was the final model (`best_tree`) trained on?

Your answer here

The entire sample of X_train_encoded was used to train the final model. Because `refit` is set to True by default in the GridSearchCV function, it would refit the model using the best hyperparameters from the grid search on the whole training set X_train_encoded.

The model can now be used to get predictions for unseen samples. For example:

[21]:
```
random_sample = X_test_encoded.sample(n=1, random_state=42)

random_sample
```

[21]:
```
      Cloud Cover_clear  Cloud Cover_cloudy  Cloud Cover_overcast  \
2005                1.0                 0.0                   0.0

      Cloud Cover_partly cloudy  Season_Autumn  Season_Spring  Season_Summer  \
2005                        0.0            0.0            0.0            0.0

      Season_Winter  Location_coastal  Location_inland  Location_mountain  \
2005            1.0               0.0              0.0                1.0

      Temperature  Humidity  Wind Speed  Precipitation (%)  \
2005         33.0      67.0         5.5                5.0

      Atmospheric Pressure  UV Index  Visibility (km)
2005               1015.42      10.0              8.0
```

[22]:
```
best_tree.predict(random_sample)
```

[22]:
```
array(['Sunny'], dtype=object)
```

5 Final thoughts

1) If you have completed this assignment in a group, please write a detailed description of how you divided the work and how you helped each other completing it:

We shared our thoughts on the questions, and after the discussion, Yanxin wrote the Python review question while Yelia handled question 2 and the Black Mirror episode. Once all the answers were written, we reviewed them together, provided feedback to one another, and revised the responses until we were both satisfied.

2) Have you used ChatGPT or a similar Large Language Model (LLM) to complete this homework? Please describe how you used the tool. **We will never deduct points for using LLMs for completing homework assignments,** but this helps us understand how you are using the tool and advise you in case we believe you are using it incorrectly.

Yes, we used ChatGPT to correct grammar. For the Black Mirror episode, we used ChatGPT to brainstorm near-future technology ideas, such as Memory Fast-Forwarder, Virtual Death Simulator, and Sensory Sharing, to help us better understand the questions related to the ethical and social implications of emerging technologies. Also, we used chatgpt to ensure the Python review answers addressed every point of the question.

3) Have you struggled with some parts (or all) of this homework? Do you have pending questions you would like to ask? Write them down here!

When we were writing the script, we struggled with a lack of inspiration and spent a lot of time trying to come up with ideas.