

Colab: <https://colab.research.google.com/drive/13BwBW8Od0iDBs1utS4MGMGuLT2Fj9zAH>

Poster

-template:

https://www.canva.com/design/DAF1ICd7R6s/qTFuQjEqSi4kMJfilO95mQ/view?utm_content=DAF1ICd7R6s&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink&mode=preview

-neu brand color: <https://brand.northeastern.edu/visual-design/color/>

Workbench VM notebook:

<https://aff482540678bd1-dot-us-central1.notebooks.googleusercontent.com/lab/tree/ml.ipynb>

latex: <https://www.overleaf.com/7121327644gdmjykkwytk>

Slides: https://docs.google.com/presentation/d/1G_kR1BK-6JMAiPoPgKHjW5Ko_I54EJ9trWsMx--cNDc/edit#slide=id.p

reference link:

Deep Neural Networks and Tabular Data: A Survey <https://arxiv.org/abs/2110.01889> (Credit: Mia)

HF tabular data Model page

https://huggingface.co/models?pipeline_tag=tabular-classification&sort=trending

https://huggingface.co/models?pipeline_tag=image-segmentation&sort=trending

<https://paperswithcode.com/method/saint> (Credit: Mia)

AI for Earth <https://microsoft.github.io/AIforEarth-Grantees/>

[firms vis](#)

[paperwithcode](#)

[elevations dataset](#)

[geo-system dataset](#)

[soil-temperature dataset](#)

[GPT paper reading tool](#)

[How to Read Research Papers \(Andrew Ng\)](#)

1. [ml-notes](#)

2. [ml-slides](#)

Canada官方Canadian Forest Fire Weather Index (FWI) System

[Canadian Forest Fire Weather Index \(FWI\) System](#)

[Canadian Wildland Fire Information System | Data Sources and Methods for Daily Maps](#)
[Canadian Wildland Fire Information System | Interactive map](#)

respond to Reviewer II:

https://docs.google.com/document/d/14y7UWV_IKcSlB0UMwODrTAz7Z-Hs7ga6qPVtgpm-aw/edit

这是poster的link

https://www.canva.com/design/DAF1x8sgRRM/9lcDg5eebObtHIU9mjvzEw/edit?utm_content=DAF1x8sgRRM&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

这是之前的slides, 我们finetune一下应该就行

https://docs.google.com/presentation/d/1G_kR1BK-6JMAiPoPgKHjW5Ko_I54EJ9trWsMx--cNDc/edit#slide=id.g294a2727d55_0_311

Results Week

WWW

A10 (December 4th):

- Response to Reviewers - Panel Review II; use the same format as before. Since most likely the review will target your simulation plan, you might need to include additional simulations; because of that, the A10 requires only the response to reviewers with a plan how will you address the situation, while the final paper will need to have those completed

Final version of your paper (December 6th EOD):

- The entire paper should be ready, and final feedback should be addressed (Panel Review I and II)
- Final analysis of results should be presented
- Conclusion should be added, including limitations and future works
- Each not addressed item from Panel Reviews will result in -1 point; please be specific in the Response to Reviewers on how did you address the comment, including the text added
- Each missing item from the list: (title, names, correct affiliation, abstract, introduction, related works, problem statement, algorithms, simulation setup/experiment design, results, conclusion with limitation and future works, references) will result in -1 point as well

Presentation (December 7th noon):

- There is no need to come to class.
- The presentation should be 15 minutes long (with a maximum of 20 minutes).
- The team's presentation video should be submitted with "watchable not downloadable" link
- You can present as a team
- There is no need to spend too much time on the algorithms. Focus on the results.
- Sample timing (no need to follow exactly):
 - Intro and motivation - 1 minute
 - Problem Description - 1-2 minutes
 - Algorithms used - 2-3 minutes (for all!)
 - Simulation setup - 2-3 minutes
 - Results - 5-7 minutes
 - Conclusion and limitations - 2 minutes
 - Future Works - 2 minutes
- [Rubric](#)
- [Links to an external site.](#)
-

Poster (December 11th EOD):

Here is a sample [poster](#)

[Links to an external site.](#)

. Since the Demo Day will be done using TVs, the best is to use 9:16 format, with 4k resolution.

Here are the grading criteria:

- Content Relevance and Impact (2 points): I will assess how well the poster addresses the question "What did you do? Why is it beneficial?" and the impact or contribution of the project
- Research and Analysis Quality (2 points): I will evaluate the depth and accuracy of research and analysis presented in the poster
- Clarity of Information (2 points): the poster should present information in a clear, concise, and understandable manner, making it easy for the audience to grasp the key points
- Visual Appeal and Design (2 points): the poster should be visually attractive, using appropriate graphics, colors, and layout to enhance the presentation and effectively communicate the message.
- Originality and Creativity (2 points): the poster should exhibit originality and creativity in both content and design, showcasing unique approaches or perspectives

Demo day (Tuesday December 12th, from 4 pm - 7 pm):

- Please arrive no later than 3 PM to set up TV. The event will start at 4:30 PM
- Be prepared to answer questions about your project during the entire event
- The event schedule is here:
<https://www.eventbrite.ca/e/student-showcase-tickets-732888508257>
- [Links to an external site.](#)
-
- You don't need to register
- We start at 4:30 with short intro about all projects - the best would be to use the video presentation you created

D

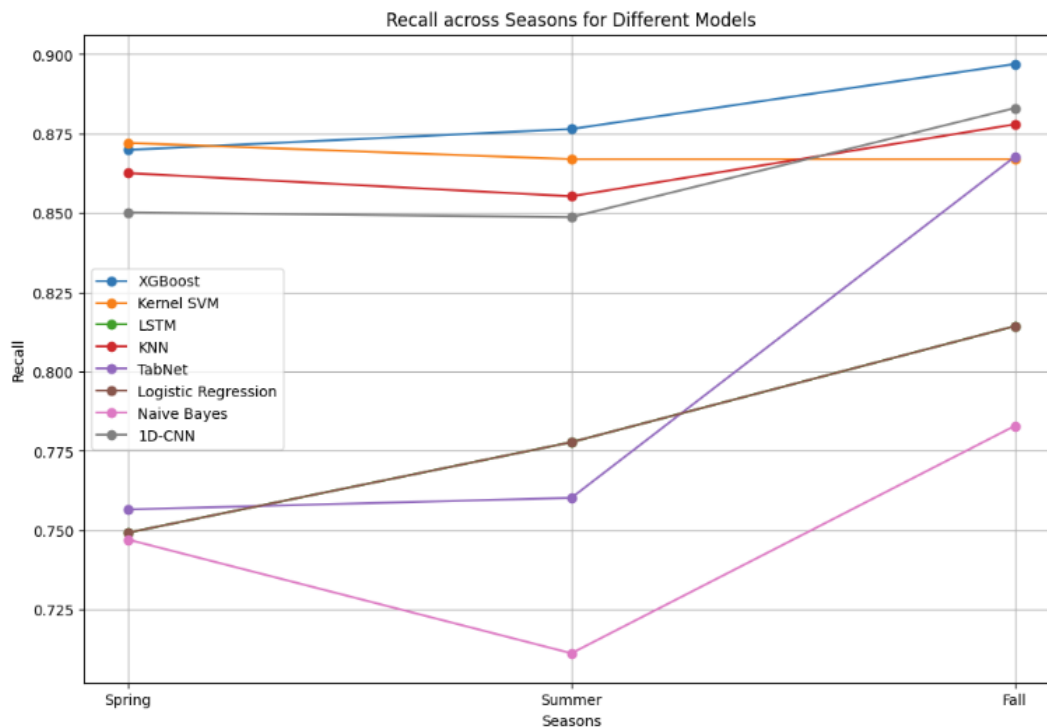
// reply to reviewer II, III

TD

Draw the model selection plow

Week11

Training



`XGB res

Fitting 3 folds for each of 10 candidates, totalling 30 fits

Model 1 - Accuracy: 0.8698830409356725

Model 1 - Recall: 0.8698830409356725

Model 1 - Confusion Matrix:

```
[[592  92]
 [ 86 598]]
```

Fitting 3 folds for each of 10 candidates, totalling 30 fits

Model 2 - Accuracy: 0.8574561403508771

Model 2 - Recall: 0.8574561403508771

Model 2 - Confusion Matrix:

```
[[561 123]
 [ 72 612]]
```

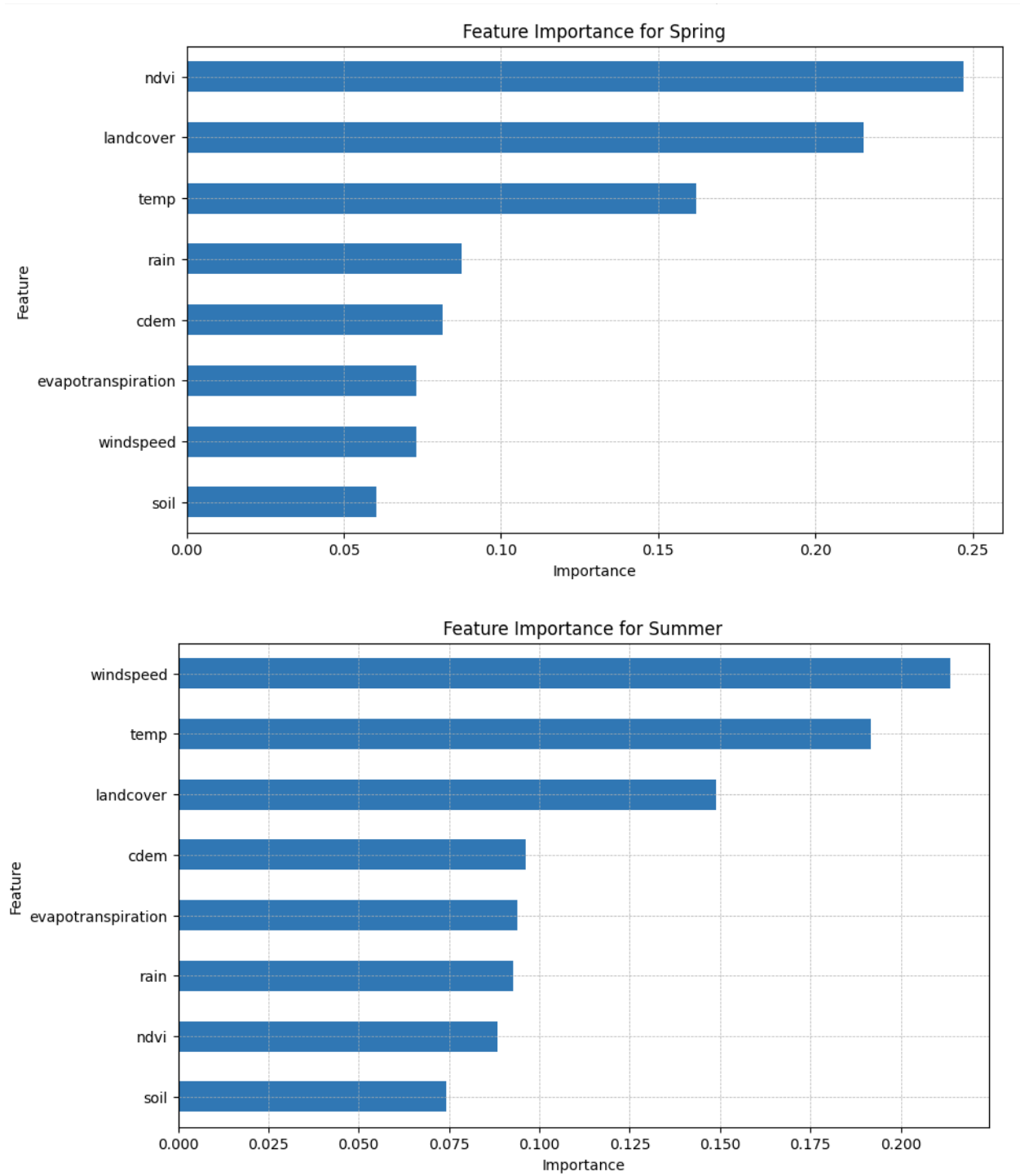
Fitting 3 folds for each of 10 candidates, totalling 30 fits

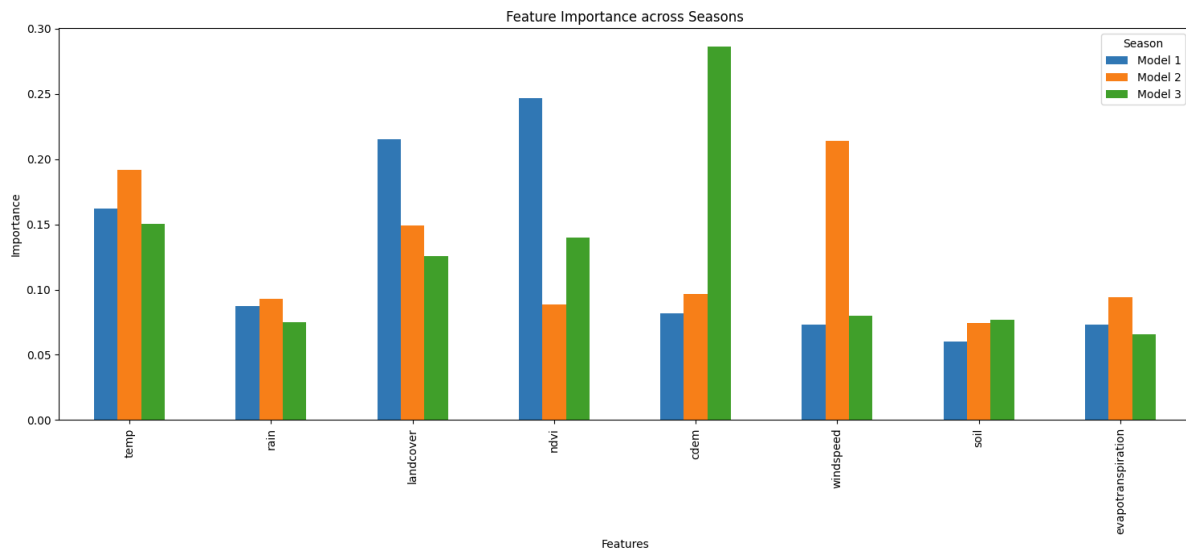
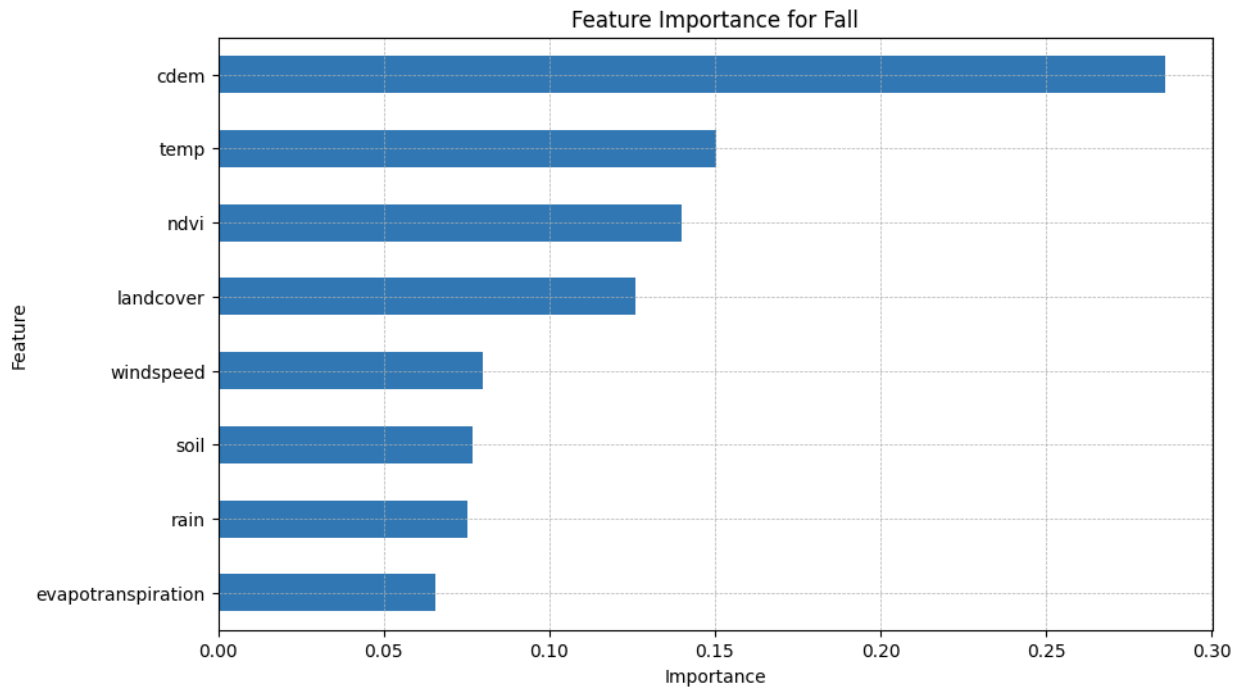
Model 3 - Accuracy: 0.8969298245614035

Model 3 - Recall: 0.8969298245614035

Model 3 - Confusion Matrix:

```
[[592  92]
 [ 49 635]]
```





SVM res

Fitting 3 folds for each of 10 candidates, totalling 30 fits

Model 1 - Accuracy: 0.8713450292397661

Model 1 - Recall: 0.871345029239766

Model 1 - F1-Score: 0.8711290764593039

Model 1 - Confusion Matrix:

```
[[568 116]
```

```

[ 60 624]]
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Model 2 - Accuracy: 0.8538011695906432
Model 2 - Recall: 0.8538011695906433
Model 2 - F1-Score: 0.8532478566463273
Model 2 - Confusion Matrix:
[[542 142]
 [ 58 626]]
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Model 3 - Accuracy: 0.8676900584795322
Model 3 - Recall: 0.8676900584795322
Model 3 - F1-Score: 0.867408853225379
Model 3 - Confusion Matrix:
[[562 122]
 [ 59 625]]

```

LSTM res

```

43/43 [=====] - 1s 3ms/step
43/43 [=====] - 2s 5ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 0s 3ms/step
Season Spring - Accuracy: 0.7573099415204678
Season Spring - Recall: 0.7573099415204678
Season Spring - F1-Score: 0.7554280397022334
Season Spring - Confusion Matrix:
[[458 226]
 [106 578]]
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 2s 5ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step

```


43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 0s 3ms/step

Season Summer - Accuracy: 0.7909356725146199

Season Summer - Recall: 0.7909356725146199

Season Summer - F1-Score: 0.7897203494771601

Season Summer - Confusion Matrix:

[[489 195]

[91 593]]

43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 4ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 1s 3ms/step
43/43 [=====] - 0s 3ms/step

Season Fall - Accuracy: 0.8347953216374269

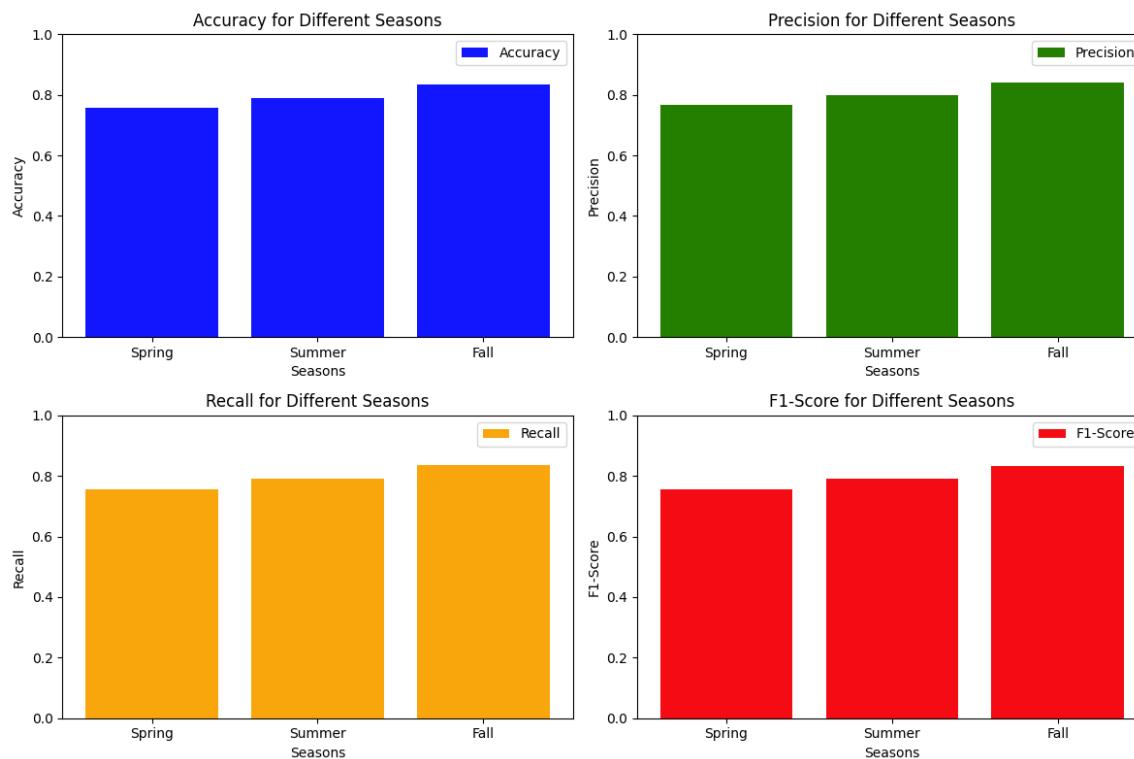
Season Fall - Recall: 0.8347953216374269

Season Fall - F1-Score: 0.8341398305175673

Season Fall - Confusion Matrix:

[[528 156]

[70 614]]



Attention res

Fitting 5 folds for each of 5 candidates, totalling 25 fits

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning:
Device used : cuda

warnings.warn(f"Device used : {self.device}")

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning:
Device used : cuda

warnings.warn(f"Device used : {self.device}")

```
epoch 0 | loss: 1.17242 | val_0_accuracy: 0.53289 | 0:00:00s
epoch 1 | loss: 0.64759 | val_0_accuracy: 0.70833 | 0:00:00s
epoch 2 | loss: 0.51453 | val_0_accuracy: 0.68129 | 0:00:00s
epoch 3 | loss: 0.47051 | val_0_accuracy: 0.69883 | 0:00:00s
epoch 4 | loss: 0.43938 | val_0_accuracy: 0.68713 | 0:00:01s
epoch 5 | loss: 0.41808 | val_0_accuracy: 0.73465 | 0:00:01s
epoch 6 | loss: 0.40781 | val_0_accuracy: 0.75658 | 0:00:01s
epoch 7 | loss: 0.38974 | val_0_accuracy: 0.73465 | 0:00:01s
epoch 8 | loss: 0.38641 | val_0_accuracy: 0.67251 | 0:00:02s
epoch 9 | loss: 0.38625 | val_0_accuracy: 0.68056 | 0:00:02s
```

```
epoch 10 | loss: 0.37799 | val_0_accuracy: 0.70395 | 0:00:02s
epoch 11 | loss: 0.37056 | val_0_accuracy: 0.69079 | 0:00:02s
epoch 12 | loss: 0.36125 | val_0_accuracy: 0.71418 | 0:00:03s
epoch 13 | loss: 0.37727 | val_0_accuracy: 0.6864 | 0:00:03s
epoch 14 | loss: 0.36309 | val_0_accuracy: 0.67032 | 0:00:03s
epoch 15 | loss: 0.36327 | val_0_accuracy: 0.63012 | 0:00:03s
epoch 16 | loss: 0.36071 | val_0_accuracy: 0.65643 | 0:00:04s
epoch 17 | loss: 0.36374 | val_0_accuracy: 0.73026 | 0:00:04s
epoch 18 | loss: 0.36233 | val_0_accuracy: 0.69371 | 0:00:04s
epoch 19 | loss: 0.35989 | val_0_accuracy: 0.72003 | 0:00:04s
epoch 20 | loss: 0.35763 | val_0_accuracy: 0.71491 | 0:00:05s
epoch 21 | loss: 0.35063 | val_0_accuracy: 0.68713 | 0:00:05s
epoch 22 | loss: 0.35668 | val_0_accuracy: 0.6652 | 0:00:05s
epoch 23 | loss: 0.35398 | val_0_accuracy: 0.69883 | 0:00:05s
epoch 24 | loss: 0.35594 | val_0_accuracy: 0.65058 | 0:00:06s
epoch 25 | loss: 0.35857 | val_0_accuracy: 0.68787 | 0:00:06s
epoch 26 | loss: 0.3586 | val_0_accuracy: 0.70833 | 0:00:06s
```

Early stopping occurred at epoch 26 with best_epoch = 6 and best_val_0_accuracy = 0.75658

Model 1 - Best Parameters: {'n_steps': 4, 'n_independent': 1, 'n_d': 64, 'n_a': 64, 'gamma': 1.5}

Model 1 - F1: 0.7495611558766629

Model 1 - Accuracy: 0.756578947368421

Model 1 - Recall: 0.756578947368421

Fitting 5 folds for each of 5 candidates, totalling 25 fits

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/callbacks.py:172: UserWarning: Best weights from best epoch are automatically used!

warnings.warn(wrn_msg)

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning:

Device used : cuda

warnings.warn(f"Device used : {self.device}")

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning:

Device used : cuda

warnings.warn(f"Device used : {self.device}")

```
epoch 0 | loss: 0.63147 | val_0_accuracy: 0.60453 | 0:00:00s
epoch 1 | loss: 0.49472 | val_0_accuracy: 0.66155 | 0:00:00s
epoch 2 | loss: 0.45887 | val_0_accuracy: 0.68787 | 0:00:00s
epoch 3 | loss: 0.42815 | val_0_accuracy: 0.70541 | 0:00:01s
epoch 4 | loss: 0.42588 | val_0_accuracy: 0.76023 | 0:00:01s
epoch 5 | loss: 0.4175 | val_0_accuracy: 0.7193 | 0:00:01s
epoch 6 | loss: 0.41588 | val_0_accuracy: 0.74561 | 0:00:01s
epoch 7 | loss: 0.40797 | val_0_accuracy: 0.74708 | 0:00:02s
epoch 8 | loss: 0.41002 | val_0_accuracy: 0.70833 | 0:00:02s
epoch 9 | loss: 0.41117 | val_0_accuracy: 0.72734 | 0:00:02s
epoch 10 | loss: 0.39814 | val_0_accuracy: 0.7098 | 0:00:02s
```

epoch 11 | loss: 0.39562 | val_0_accuracy: 0.7076 | 0:00:02s
epoch 12 | loss: 0.39992 | val_0_accuracy: 0.73319 | 0:00:03s
epoch 13 | loss: 0.3908 | val_0_accuracy: 0.74927 | 0:00:03s
epoch 14 | loss: 0.38243 | val_0_accuracy: 0.74708 | 0:00:03s
epoch 15 | loss: 0.38637 | val_0_accuracy: 0.72442 | 0:00:03s
epoch 16 | loss: 0.38829 | val_0_accuracy: 0.7405 | 0:00:03s
epoch 17 | loss: 0.38181 | val_0_accuracy: 0.74635 | 0:00:04s
epoch 18 | loss: 0.38467 | val_0_accuracy: 0.74342 | 0:00:04s
epoch 19 | loss: 0.38101 | val_0_accuracy: 0.73684 | 0:00:04s
epoch 20 | loss: 0.37878 | val_0_accuracy: 0.71272 | 0:00:04s
epoch 21 | loss: 0.37802 | val_0_accuracy: 0.73319 | 0:00:04s
epoch 22 | loss: 0.38118 | val_0_accuracy: 0.74854 | 0:00:05s
epoch 23 | loss: 0.37672 | val_0_accuracy: 0.74488 | 0:00:05s
epoch 24 | loss: 0.37827 | val_0_accuracy: 0.71857 | 0:00:05s

Early stopping occurred at epoch 24 with best_epoch = 4 and best_val_0_accuracy = 0.76023

Model 2 - Best Parameters: {'n_steps': 3, 'n_independent': 1, 'n_d': 8, 'n_a': 16, 'gamma': 1.3}

Model 2 - F1: 0.7579805825242718

Model 2 - Accuracy: 0.7602339181286549

Model 2 - Recall: 0.760233918128655

Fitting 5 folds for each of 5 candidates, totalling 25 fits

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/callbacks.py:172: UserWarning: Best weights from best epoch are automatically used!

warnings.warn(wrn_msg)

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning: Device used : cuda

warnings.warn(f"Device used : {self.device}")

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/abstract_model.py:82: UserWarning: Device used : cuda

warnings.warn(f"Device used : {self.device}")

epoch 0 | loss: 1.01798 | val_0_accuracy: 0.50365 | 0:00:00s
epoch 1 | loss: 0.60465 | val_0_accuracy: 0.50219 | 0:00:00s
epoch 2 | loss: 0.52282 | val_0_accuracy: 0.49708 | 0:00:01s
epoch 3 | loss: 0.43521 | val_0_accuracy: 0.58991 | 0:00:01s
epoch 4 | loss: 0.43483 | val_0_accuracy: 0.55702 | 0:00:01s
epoch 5 | loss: 0.40504 | val_0_accuracy: 0.59576 | 0:00:02s
epoch 6 | loss: 0.39141 | val_0_accuracy: 0.70906 | 0:00:02s
epoch 7 | loss: 0.3748 | val_0_accuracy: 0.63962 | 0:00:02s
epoch 8 | loss: 0.35465 | val_0_accuracy: 0.64985 | 0:00:03s
epoch 9 | loss: 0.33944 | val_0_accuracy: 0.71345 | 0:00:03s
epoch 10 | loss: 0.33859 | val_0_accuracy: 0.67909 | 0:00:03s
epoch 11 | loss: 0.33896 | val_0_accuracy: 0.6864 | 0:00:04s
epoch 12 | loss: 0.32869 | val_0_accuracy: 0.66886 | 0:00:04s
epoch 13 | loss: 0.34076 | val_0_accuracy: 0.69518 | 0:00:04s

| | | | |
|----------|---------------|-------------------------|----------|
| epoch 14 | loss: 0.33627 | val_0_accuracy: 0.72734 | 0:00:05s |
| epoch 15 | loss: 0.32987 | val_0_accuracy: 0.72442 | 0:00:05s |
| epoch 16 | loss: 0.32888 | val_0_accuracy: 0.76608 | 0:00:06s |
| epoch 17 | loss: 0.33765 | val_0_accuracy: 0.74415 | 0:00:06s |
| epoch 18 | loss: 0.33327 | val_0_accuracy: 0.74488 | 0:00:06s |
| epoch 19 | loss: 0.339 | val_0_accuracy: 0.73173 | 0:00:07s |
| epoch 20 | loss: 0.32382 | val_0_accuracy: 0.73099 | 0:00:07s |
| epoch 21 | loss: 0.32562 | val_0_accuracy: 0.72515 | 0:00:07s |
| epoch 22 | loss: 0.33315 | val_0_accuracy: 0.75073 | 0:00:08s |
| epoch 23 | loss: 0.31806 | val_0_accuracy: 0.76535 | 0:00:08s |
| epoch 24 | loss: 0.32041 | val_0_accuracy: 0.69225 | 0:00:08s |
| epoch 25 | loss: 0.32128 | val_0_accuracy: 0.74781 | 0:00:09s |
| epoch 26 | loss: 0.3197 | val_0_accuracy: 0.78289 | 0:00:09s |
| epoch 27 | loss: 0.32578 | val_0_accuracy: 0.76462 | 0:00:10s |
| epoch 28 | loss: 0.32386 | val_0_accuracy: 0.78143 | 0:00:10s |
| epoch 29 | loss: 0.32114 | val_0_accuracy: 0.80409 | 0:00:11s |
| epoch 30 | loss: 0.3274 | val_0_accuracy: 0.77851 | 0:00:11s |
| epoch 31 | loss: 0.30991 | val_0_accuracy: 0.77851 | 0:00:11s |
| epoch 32 | loss: 0.31934 | val_0_accuracy: 0.7617 | 0:00:12s |
| epoch 33 | loss: 0.31969 | val_0_accuracy: 0.76462 | 0:00:12s |
| epoch 34 | loss: 0.31533 | val_0_accuracy: 0.78216 | 0:00:13s |
| epoch 35 | loss: 0.31097 | val_0_accuracy: 0.79167 | 0:00:13s |
| epoch 36 | loss: 0.31216 | val_0_accuracy: 0.78289 | 0:00:14s |
| epoch 37 | loss: 0.32277 | val_0_accuracy: 0.73538 | 0:00:14s |
| epoch 38 | loss: 0.31392 | val_0_accuracy: 0.78801 | 0:00:14s |
| epoch 39 | loss: 0.31783 | val_0_accuracy: 0.77632 | 0:00:15s |
| epoch 40 | loss: 0.31524 | val_0_accuracy: 0.8114 | 0:00:15s |
| epoch 41 | loss: 0.30909 | val_0_accuracy: 0.8136 | 0:00:15s |
| epoch 42 | loss: 0.30939 | val_0_accuracy: 0.80775 | 0:00:16s |
| epoch 43 | loss: 0.30019 | val_0_accuracy: 0.7902 | 0:00:16s |
| epoch 44 | loss: 0.30537 | val_0_accuracy: 0.80702 | 0:00:16s |
| epoch 45 | loss: 0.31026 | val_0_accuracy: 0.82529 | 0:00:17s |
| epoch 46 | loss: 0.31071 | val_0_accuracy: 0.82018 | 0:00:17s |
| epoch 47 | loss: 0.32128 | val_0_accuracy: 0.81067 | 0:00:17s |
| epoch 48 | loss: 0.31585 | val_0_accuracy: 0.83041 | 0:00:18s |
| epoch 49 | loss: 0.32436 | val_0_accuracy: 0.8231 | 0:00:18s |
| epoch 50 | loss: 0.32871 | val_0_accuracy: 0.81287 | 0:00:18s |
| epoch 51 | loss: 0.32144 | val_0_accuracy: 0.80921 | 0:00:19s |
| epoch 52 | loss: 0.32103 | val_0_accuracy: 0.7924 | 0:00:19s |
| epoch 53 | loss: 0.31678 | val_0_accuracy: 0.80702 | 0:00:19s |
| epoch 54 | loss: 0.31723 | val_0_accuracy: 0.81871 | 0:00:20s |
| epoch 55 | loss: 0.31302 | val_0_accuracy: 0.83114 | 0:00:20s |
| epoch 56 | loss: 0.30105 | val_0_accuracy: 0.8348 | 0:00:20s |
| epoch 57 | loss: 0.30187 | val_0_accuracy: 0.82456 | 0:00:21s |

| | | | |
|----------|---------------|-------------------------|----------|
| epoch 58 | loss: 0.30166 | val_0_accuracy: 0.8114 | 0:00:21s |
| epoch 59 | loss: 0.30238 | val_0_accuracy: 0.81287 | 0:00:21s |
| epoch 60 | loss: 0.3006 | val_0_accuracy: 0.81652 | 0:00:22s |
| epoch 61 | loss: 0.30726 | val_0_accuracy: 0.81725 | 0:00:22s |
| epoch 62 | loss: 0.30973 | val_0_accuracy: 0.82968 | 0:00:22s |
| epoch 63 | loss: 0.31725 | val_0_accuracy: 0.81798 | 0:00:23s |
| epoch 64 | loss: 0.31126 | val_0_accuracy: 0.82529 | 0:00:23s |
| epoch 65 | loss: 0.30388 | val_0_accuracy: 0.83699 | 0:00:24s |
| epoch 66 | loss: 0.30622 | val_0_accuracy: 0.83114 | 0:00:24s |
| epoch 67 | loss: 0.30517 | val_0_accuracy: 0.82091 | 0:00:25s |
| epoch 68 | loss: 0.30562 | val_0_accuracy: 0.81579 | 0:00:25s |
| epoch 69 | loss: 0.29915 | val_0_accuracy: 0.83333 | 0:00:25s |
| epoch 70 | loss: 0.31296 | val_0_accuracy: 0.80263 | 0:00:26s |
| epoch 71 | loss: 0.29921 | val_0_accuracy: 0.81652 | 0:00:26s |
| epoch 72 | loss: 0.30009 | val_0_accuracy: 0.82895 | 0:00:27s |
| epoch 73 | loss: 0.3084 | val_0_accuracy: 0.81871 | 0:00:27s |
| epoch 74 | loss: 0.28937 | val_0_accuracy: 0.82091 | 0:00:28s |
| epoch 75 | loss: 0.30695 | val_0_accuracy: 0.8231 | 0:00:28s |
| epoch 76 | loss: 0.30545 | val_0_accuracy: 0.84503 | 0:00:28s |
| epoch 77 | loss: 0.30498 | val_0_accuracy: 0.83406 | 0:00:29s |
| epoch 78 | loss: 0.29937 | val_0_accuracy: 0.85746 | 0:00:29s |
| epoch 79 | loss: 0.29703 | val_0_accuracy: 0.84649 | 0:00:29s |
| epoch 80 | loss: 0.29497 | val_0_accuracy: 0.85015 | 0:00:30s |
| epoch 81 | loss: 0.29616 | val_0_accuracy: 0.8443 | 0:00:30s |
| epoch 82 | loss: 0.29583 | val_0_accuracy: 0.84357 | 0:00:30s |
| epoch 83 | loss: 0.29415 | val_0_accuracy: 0.84795 | 0:00:31s |
| epoch 84 | loss: 0.30269 | val_0_accuracy: 0.84503 | 0:00:31s |
| epoch 85 | loss: 0.31031 | val_0_accuracy: 0.84868 | 0:00:31s |
| epoch 86 | loss: 0.30107 | val_0_accuracy: 0.85307 | 0:00:32s |
| epoch 87 | loss: 0.30127 | val_0_accuracy: 0.86111 | 0:00:32s |
| epoch 88 | loss: 0.29804 | val_0_accuracy: 0.86404 | 0:00:32s |
| epoch 89 | loss: 0.29621 | val_0_accuracy: 0.85307 | 0:00:33s |
| epoch 90 | loss: 0.30561 | val_0_accuracy: 0.85673 | 0:00:33s |
| epoch 91 | loss: 0.3075 | val_0_accuracy: 0.86038 | 0:00:33s |
| epoch 92 | loss: 0.30669 | val_0_accuracy: 0.85453 | 0:00:34s |
| epoch 93 | loss: 0.30794 | val_0_accuracy: 0.84576 | 0:00:34s |
| epoch 94 | loss: 0.31386 | val_0_accuracy: 0.85892 | 0:00:34s |
| epoch 95 | loss: 0.32064 | val_0_accuracy: 0.86477 | 0:00:35s |
| epoch 96 | loss: 0.32323 | val_0_accuracy: 0.85673 | 0:00:35s |
| epoch 97 | loss: 0.3186 | val_0_accuracy: 0.8655 | 0:00:35s |
| epoch 98 | loss: 0.32098 | val_0_accuracy: 0.86769 | 0:00:36s |
| epoch 99 | loss: 0.31186 | val_0_accuracy: 0.85673 | 0:00:36s |

Stop training because you reached max_epochs = 100 with best_epoch = 98 and best_val_0_accuracy = 0.86769

/usr/local/lib/python3.10/dist-packages/pytorch_tabnet/callbacks.py:172: UserWarning: Best weights from best epoch are automatically used!

warnings.warn(wrn_msg)

Model 3 - Best Parameters: {'n_steps': 5, 'n_independent': 2, 'n_d': 64, 'n_a': 32, 'gamma': 1.5}

Model 3 - F1: 0.8669648935135661

Model 3 - Accuracy: 0.8676900584795322

Model 3 - Recall: 0.8676900584795322

Simulation

WWW

For next week, please prepare for the next panel review. Here are some bullet points:

Clean up the paper

The algorithm part should be completed and each section should indicate which student completed what algorithm

Make sure the simulation scenarios are clear

Apply feedback from previous round of review

For the Panel Review II you will submit three things:

paper

response to reviewers (if you need to adjust it, please do, otherwise same as before)

the simulation flowchart that we did today in the class (include it as a separate file)

Here are some good papers with simulation setup as a sample to see:

Finally, you can get 3 bonus points as a group if you'll prepare a short, less than a minute, promotional video about your project. It should include a title slide with Northeastern logo and brand colorsLinks to an external site.. Talk 10-15 seconds about the motivation of the project, then 30-40 about the solutions and why they make an impact. Finish with logo of Northeastern in ending slide, where you can also include the names of people involved. Submission deadline is November 30th, over an email to me. You can include your LinkedIn profile links as it will be posted on social media.

No class next week, but the work stands! I will see you on November 30th to discuss your results!

-notes(from Huizi)

- Layout detail objective: feature importance between seasons
- Specify what models used
- Applied which deep learning models and how customized
- How many time simulation run? what parameters were adjusted
- Add hyperparameter tuning
- Histogram little more details
- Include in panel review & submit separately

TD

//Clean up the paper

//The algorithm part should be completed and each section should indicate which student completed what algorithm

Make sure the simulation scenarios are clear

//Apply feedback from previous round of review

//Migrate to new server - to github/colab

//flowchart

REP

bug

```

46 dataset = dataset.map(parse_tfrecord, num_parallel_calls=5)
47 dataset = dataset.map(to_tuple, num_parallel_calls=5)
48 return dataset

1 dataset_spring = get_dataset(f'{TRAINING_OUTPUT_PATH}Spring')
2 test_balance_dataset(dataset_spring)
3 dataset_spring

gs://wildfire-export/training_export/Spring*
-----
PermissionDeniedError                                Traceback (most recent call last)
<ipython.input-41-32e213b6dfc1> in <cell line: 2>()
      1 dataset_spring = get_dataset(f'{TRAINING_OUTPUT_PATH}Spring')
----> 2 test_balance_dataset(dataset_spring)
      3 dataset_spring

↳ 5 frames
/usr/local/lib/python3.10/dist-packages/tensorflow/python/framework/ops.py in raise_from_not_ok_status(e, name)
5886 def raise_from_not_ok_status(e, name):
5887     e.message += '[name: %s]' % str(name if name is not None else '')
-> 5888     raise core._status_to_exception(e) from None # pylint: disable=protected-access
5889
5890
PermissionDeniedError: (function_node wrapped_iteratorGetNext output_types 2 device /job:localhost/replica:0/task:0/device:CPU:0) Error executing an HTTP request: HTTP response code 403 with body '<?xml version='1.0' encoding='UTF-8'?><Error>
<Code>UserProjectAccountProblem</Code><Message>The project to be billed is associated with an absent billing account.</Message><Details>The billing account for the owning project is disabled in state absent</Details></Error>'

```



```

dataset_spring = get_dataset(f'{TRAINING_OUTPUT_PATH}Spring')
test_balance_dataset(dataset_spring)

gs://wildfire-export/training_export/Spring*
2023-11-21 03:03:21.093316: W tensorflow/tsl/platform/cloud/google_auth_provider.cc:184] All attempts to get a Google authentication bearer token failed, returning an empty token. Retrieving token from files failed with "NO
T FOUND: Could not locate the credentials file.". Retrieving token from GCE failed with "ABORTED: All 10 retry attempts failed. The last failure: Error executing an HTTP request: HTTP response code 503 with body '<DOCTYPE
html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html><head>
<meta type="copyright" content="Copyright (C) 1996-2016 The Squid Software Foundation and contributors">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>ERROR: The requested URL could not be retrieved</title>
<style type="text/css"><!--
/*
* Copyright (C) 1996-2016 The Squid Software Foundation and contributors
*
* Squid software is distributed under GPLv2+ license".
-----
FailedPreconditionError                                Traceback (most recent call last)
Cell In[41], line 1
----> 1 dataset_spring = get_dataset(f'{TRAINING_OUTPUT_PATH}Spring')
      2 test_balance_dataset(dataset_spring)

Cell In[40], line 44, in get_dataset(pattern)
    36 """Function to read, parse and format to tuple a set of input tfrecord files.
    37 Get all the files matching the pattern, parse and convert to tuple.
    38 Args:
    (...)
    41 A tf.data.Dataset
    42 """
    43 print(pattern)
----> 44 glob = tf.io.gfile.glob(pattern)
    45 dataset = tf.data.TFRecordDataset(glob, compression type='GZIP')
    46 dataset = dataset.map(parse_tfrecord, num_parallel_calls=5)

File ~/work/van-speech-nlp/condaENV/lib/python3.8/site-packages/tensorflow/python/lib/io/file_io.py:441, in get_matching_files_v2(pattern)
    387 r"""Returns a list of files that match the given pattern(s).
    388
    389 The patterns are defined as strings. Supported patterns are defined
    (...)
    435 errors.NotFoundError: If pattern to be matched is an invalid directory.
    436 """
    437 if isinstance(pattern, six.string_types):
    438     return [
    439         # Convert the filenames to string from bytes.
    440         compat.as_str_any(matching_filename)
    441         for matching_filename in pywrap_file_io.GetMatchingFiles(
    442             compat.as_bytes(pattern))
    443     ]
    444 else:
    445     return [
    446         # Convert the filenames to string from bytes.
    447         compat.as_str_any(matching_filename) # pylint: disable=g-complex-comprehension
    (...)
    450         compat.as_bytes(single_filename))
    451 ]

FailedPreconditionError: Error executing an HTTP request: libcurl code 77 meaning 'Problem with the SSL CA cert (path? access rights?)', error details: error setting certificate verify locations: CAfile: /etc/ssl/certs/ca-
certificates.crt CApath: none
when reading gs://wildfire-export/training_export

```

Algorithm IV Algorithm V

WWW

Reply feedback + algo

For this week, please continue to work on the algorithms. Implementation should be done by November 15th, so we can start running some simulations.

Next week, we'll be working on designing the experiments. Please read the article below:

<https://www.baeldung.com/cs/research-design-simulation-experiments>

Justin

TD

implement/study feature importance for NN

//-resolve previous comment about more detail workflow

//train/val split vs CV

<https://stackoverflow.com/questions/72361384/is-k-folds-cross-validation-a-smarter-idea-than-using-a-validation-set-instead>

Transformer -image pipeline

// refine test pipeline

//implement <https://arxiv.org/abs/1908.07442> tabformer random search

//- TabNet : Attentive Interpretable Tabular Learning

- TabTransformer

//-finetuning LSTM:

// using random search

// lstm feature importance

// read feature importance for LSTM/attention

Bug

```
UnknownError: Graph execution error:  
  
Fail to find the dnn implementation.  
[[{{node CudnnRNN}}]]  
[[sequential_1/lstm/PartitionedCall]] [Op:__inference_train_function_56595]
```

-again this error, change to TF2.12 kernel, probably is due to tf version

Algorithm III

WWW

Justin

TD

-read feature importance for LSTM/attention

coding->paper->ppt->recording

-finish LSTM+attention part in paper?

-finetuning LSTM

-attention to table data?

<https://towardsdatascience.com/transformers-for-tabular-data-tabtransformer-deep-dive-5fb2438da820>

- TabNet : Attentive Interpretable Tabular Learning

- TabTransformer

-image pipeline

AC

-migrate to new notebook due to bug and budget issue

-finish my part of slides draft

- finish my part of paper XGBOOST + randomized search section
- resolved 2 TA review comments
- finish prediction data transformation, get test prediction working for table data
- random search for XGB training, max_search120, test accuracy

```

Model 1 - Accuracy: 0.8108333333333333
Model 1 - Recall: 0.8108333333333333
Model 1 - Confusion Matrix:
[[979 221]
 [233 967]]
Model 2 - Accuracy: 0.8154166666666667
Model 2 - Recall: 0.8154166666666667
Model 2 - Confusion Matrix:
[[ 940 260]
 [ 183 1017]]
Model 3 - Accuracy: 0.77125
Model 3 - Recall: 0.77125
Model 3 - Confusion Matrix:
[[1023 177]
 [ 372 828]]

```

- random search for kernel SVM, max_iter=10
- result is around ~80

Bug

- Fixing this error while fitting NN

```

UnknownError: Graph execution error:

Detected at node CudnnRNN defined at (most recent call last):
<stack traces unavailable>
Fail to find the dnn implementation.
[[{{node CudnnRNN}}]]
[[sequential_2/lstm/PartitionedCall]] [Op: inference_train_function_75043]

```

- kernel died again while training NN, new bug after kernel restarting while training NN
- kernel died randomly while opened for long time, need to reinstall lots of packages

Mia

TODO:

1. Data processing
 - a. Limit our scope in west Canada (especially test data)
 - ~~b. Make sure training data contains similar number of two classes after splitting evaluation data~~
 - ~~c. Increase training amount (3600 per season -> 7200 per season)~~
 - ~~d. Increase test amount (240 per season -> 800 per season)~~
2. Improve baseline ann
 - ~~a. Make it deeper~~

- i. Improved but not much
- b. Search for mature solutions for tabular data
 - i. https://colab.research.google.com/github/keras-team/keras-io/blob/master/examples/structured_data/ipynb/tabtransformer.ipynb
Only add embedding & attention to categorical columns while ours are numerical so it doesn't help
 - ii. [releaunifreiburg/WellTunedSimpleNets: \[NeurIPS 2021\] Well-tuned Simple Nets Excel on Tabular Datasets](#)
- c. Fine-tune hyperparameters
 - i. [Getting started with KerasTuner](#)
 - ii. [CSC321 Lecture 21: Bayesian Hyperparameter Optimization](#)
- ~~3. Inspect metrics performance on test data~~
- ~~4. Feature importance analysis in ann~~
 - a. [Feature Importance Chart in neural network using Keras in Python - Stack Overflow](#)
 - b. [LSTM Feature Importance | Kaggle](#)

Observation:

Before I increased the training dataset, the shallow ANN always outperformed the deeper one. But after I doubled the amount of training samples, the shallow one can't compete with the deeper one.

Algorithm II

WWW: implement algorithm

Mia

-issue

Accuracy always show 0 when using `tf.keras.metrics.Accuracy()`. I checked this but it doesn't apply to my situation. If I change the class to string 'accuracy' it works. I finally used `tf.keras.metrics.BinaryAccuracy()` to solve it.

<https://stackoverflow.com/questions/65722752/neural-network-accuracy-is-always-0-while-training-classification-problem-in-ker>

-Feature importance

[Methods for interpreting and understanding deep neural networks - ScienceDirect](#)
[8.5 Permutation Feature Importance | Interpretable Machine Learning](#)

-improve baseline ann

[Structured data learning with TabTransformer](#)

Justin

-ac:

-kernel svm

```
Model 3 - Accuracy: 0.8245614035087719
Model 3 - Recall: 0.8245614035087718
Model 3 - f1: 0.8240740740740742
Model 3 - Confusion Matrix:
[[264  78]
 [ 42 300]]
Model 3 - Accuracy: 0.8084795321637427
Model 3 - Recall: 0.8084795321637427
Model 3 - f1: 0.8079767525089416
Model 3 - Confusion Matrix:
[[259  83]
 [ 48 294]]
Model 3 - Accuracy: 0.8669590643274854
Model 3 - Recall: 0.8669590643274854
Model 3 - f1: 0.8667194843848698
Model 3 - Confusion Matrix:
[[282  60]
 [ 31 311]]
```

-lstm

```
Training Season Spring: 100%|██████████| 10/10 [01:06<00:00, 6.61s/it]
22/22 [=====] - 1s 3ms/step
Season Spring - Accuracy: 0.8084795321637427
Season Spring - Recall: 0.8084795321637427
Season Spring - F1-Score: 0.8071402281928599
Season Spring - Confusion Matrix:
[[248  94]
 [ 37 305]]
```

```
Training Season Summer: 100%|██████████| 10/10 [01:20<00:00, 8.07s/it]
22/22 [=====] - 1s 3ms/step
Season Summer - Accuracy: 0.8099415204678363
Season Summer - Recall: 0.8099415204678362
Season Summer - F1-Score: 0.8094135802469136
Season Summer - Confusion Matrix:
[[259  83]
 [ 47 295]]
```

```
Training Season Fall: 100%|██████████| 10/10 [01:28<00:00, 8.85s/it]
22/22 [=====] - 1s 3ms/step
Season Fall - Accuracy: 0.868421052631579
Season Fall - Recall: 0.868421052631579
Season Fall - F1-Score: 0.868200191833379
Season Fall - Confusion Matrix:
[[283  59]
 [ 31 311]]
```

-knn

```
print(f"Season {season} - Confusion Matrix:\n{conf_matrix}")
```

```
Season Spring - Accuracy: 0.8523391812865497
Season Spring - Recall: 0.8523391812865497
Season Spring - F1-Score: 0.852268134490656
Season Spring - Confusion Matrix:
[[284  58]
 [ 43 299]]
Season Summer - Accuracy: 0.8216374269005848
Season Summer - Recall: 0.8216374269005848
Season Summer - F1-Score: 0.8213380378185804
Season Summer - Confusion Matrix:
[[267  75]
 [ 47 295]]
Season Fall - Accuracy: 0.8654970760233918
Season Fall - Recall: 0.8654970760233918
Season Fall - F1-Score: 0.8651234567901236
Season Fall - Confusion Matrix:
[[278  64]
 [ 28 314]]
```

- xgboost

```
Model 1 - Accuracy: 0.8971291866028708
Model 1 - Recall: 0.8971291866028708
Model 2 - Accuracy: 0.871191135734072
Model 2 - Recall: 0.871191135734072
Model 3 - Accuracy: 0.9166666666666666
Model 3 - Recall: 0.9166666666666666
```

```
for model_num, importances in season_feature_importances.items():
    print(f"Feature Importances for {model_num}:\n")
    for feature, importance in zip(train_dataframe_X.columns, importances):
        print(f"{feature}: {importance}")
    print("\n")
```

Feature Importances for Model 1:

```
temp: 0.2531953454017639
rain: 0.09257130324840546
landcover: 0.19848620891571045
ndvi: 0.150830939412117
cdem: 0.07563601434230804
windspeed: 0.08199838548898697
soil: 0.06815371662378311
evapotranspiration: 0.07912809401750565
```

Feature Importances for Model 2:

```
temp: 0.19929903745651245
rain: 0.09104562550783157
landcover: 0.14824318885803223
ndvi: 0.08415516465902328
cdem: 0.15330570936203003
windspeed: 0.15894167125225067
soil: 0.06981781870126724
evapotranspiration: 0.0951918289065361
```

-define years: code is running, but get this bug while doing stratifiedSample, going to fix this tomorrow

-define input features in feature section

I am going to add more features in future, e.g. population. but for now its enough
























- add docs for onboarding
- start testing seasonal code

-bug

Why are some labels have value 2??

| | Features | Labels |
|------|---|--------|
| 0 | [-0.046000004, -0.971, 0.125, -0.20523304, -0.... | 0.0 |
| 1 | [-2.7142859, -1.7, 1.0, -2.956044, -0.9267423,... | 0.0 |
| 2 | [0.052000046, -0.944, -1.0, 0.49949956, -0.830... | 2.0 |
| 3 | [0.10542858, -0.975, 0.0, 0.2807225, -0.872285... | 2.0 |
| 4 | [0.06628573, -0.975, -0.125, 0.27272725, -0.89... | 2.0 |
| ... | ... | ... |
| 2425 | [-0.4185714, -0.98, 0.125, -0.4993447, -0.9134... | 0.0 |
| 2426 | [0.04057145, -0.961, -1.0, 0.61067057, -0.4353... | 2.0 |
| 2427 | [0.06457138, -0.966, -1.0, 0.5011444, -0.50761... | 2.0 |
| 2428 | [-0.07085717, -0.937, 0.25, 0.35336196, -0.815... | 0.0 |
| 2429 | [-0.076571405, -0.94, 0.0, -0.01167467, -0.82... | 0.0 |

Output takes several hours, maybe after data is exported, someone could take a look and see if data is valid? Would have to work on my other project which DDL is next Monday. Feel free to ping me if need anything.

| | | | |
|---|----------|---|-------|
| > | Fall-8 |  | <1m |
| > | Fall-7 |  | <1m |
| > | Fall-6 |  | <1m |
| > | Fall-5 |  | <1m |
| > | Fall-4 |  | <1m |
| > | Fall-3 |  | <1m |
| > | Fall-2 |  | <1m |
| > | Fall-1 |  | <1m |
| > | Fall-0 |  | 15m |
| > | Summer-8 |  | 35m |
| > | Summer-7 |  | 36m |
| > | Summer-6 |  | 37m |
| > | Summer-5 |  | ✓ 46m |
| > | Summer-4 |  | ✓ 29m |
| > | Summer-3 |  | ✓ 40m |
| > | Summer-2 |  | ✓ 1h |
| > | Summer-1 |  | ✓ 56m |
| > | Summer-0 |  | ✓ 56m |
| > | Spring-8 |  | ✓ 2h |
| > | Spring-7 |  | ✓ 41m |
| > | Spring-6 |  | ✓ 54m |
| > | Spring-5 |  | ✓ 28m |
| > | Spring-4 |  | ✓ 46m |

Train data is exporting in gee, going to figure out if sampling 27 image only 1 image contain non-zero element in fire bands is normal or not.

```

tdd
121: def test_fire():
    Map = geemap.Map(center=[47.295, -119.086], zoom=6)

    # Create an empty list to store the test results
    test_results = []

    for season in combine_image_list:
        for image_info in season:
            test_img = image_info
            t21_test = test_img.select('is_fire')

            region = ee.Geometry.Point([-122.084, 37.422]).buffer(10000)
            num_samples = 1000

            samples = t21_test.sample(region=region, scale=30, numPixels=num_samples)
            sample_values = samples.aggregate_array('is_fire')
            lst = sample_values.getInfo()

            contains_non_zero = any(element != 0 for element in lst)

            test_results.append(contains_non_zero)

    print(test_results)

test_fire()
[False, False, False, False, False, False, False, False, False, False, False, False, False, False, False, False, False, False, True, False, False, False, False]

```

Actively working on this

Spring-2

ID: 34I3LH5GEYMCTP7CXXNRSSC6

Phase: **Failed**

Runtime: **2s** (started 2023-10-21 07:13:18 +0000)

Attempted 1 time

Error: Image.stratifiedSample: The class band must be integer typed. (Error code: 3)

Spring-1

ID: JOTNDAQRRKZHGSMRBLBOP7QC

Phase: **Failed**

Runtime: **0s** (started 2023-10-21 04:06:18 +0000)

Attempted 1 time

Error: Image.unmask: If one image has no bands, the other must also have no bands. Got 0 and 1. (Error code: 3)

-one of image is missing bands

-autocomplete in jupyter

<https://www.google.com/search?q=enable+autocomplete+in+jupyter+notebook>

Algorithm I

www: implement pipeline, look into algorithm

Justin:

Metric:

IOU <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>

MIOU <https://medium.com/@cyborg.team.nitr/miou-calculation-4875f918f4cb>

Model:

XGBOOST: input tabula data

https://huggingface.co/models?pipeline_tag=image-segmentation&sort=trending

Input: image

1. SegFormer (b0-sized) model fine-tuned on ADE20k

<https://huggingface.co/nvidia/segformer-b0-finetuned-ade-512-512>

Enhancing AI Segmentation Models for Autonomous Vehicle Safety - NVIDIA DRIVE Labs Ep.

28 <https://www.youtube.com/watch?v=nBjXyoltCHU>

DeepInsight: <https://zhuanlan.zhihu.com/p/379054782>

-bug list:

-workbench collaboration feature

<https://stackoverflow.com/questions/71888324/is-real-time-collaboration-supported-by-user-managed-notebooks-on-google-vertex-ai?text=Under%20%22Metadata%22%2C%20click%20%22.to%20use%20the%20collaboration%20feature>.

-Install packages in workbench IAM

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
ERROR conda.core.link_execute(945): An error occurred while uninstalling package 'conda-forge/linux-64::conda-23.7.4-py310hff52083_0'.
Rolling back transaction: done
```

```
[Errno 13] Permission denied: '/opt/conda/lib/python3.10/site-packages/conda-23.7.4.dist-info/INSTALLER' -> '/opt/conda/lib/python3.10/site-packages/conda-23.7.4.dist-info/INSTALLER.c~'
()
```

<https://cloud.google.com/vertex-ai/docs/workbench/user-managed/dependencies>

-conda + terminal

-NEU discovery cluster for training

-**EEException**: Request had insufficient authentication scopes.

<https://gis.stackexchange.com/questions/447410/how-to-fix-eeexception-request-had-insufficient-authentication-scopes-in-gee>

-vm not responding

Features

Region

To get target regions

[FAO GAUL: Global Administrative Unit Layers 2015. First-Level Administrative Units | Earth Engine Data Catalog | Google for Developers](#)

```
ee.FeatureCollection('FAO/GAUL/2015/level1').filter(ee.Filter.eq('ADM0_NAME','Canada')).distinct('ADM1_NAME')
```

```

▼ 0: Feature 00010000000000000008 (Polygon, 10 properties)
  type: Feature
  id: 00010000000000000008
  ▸ geometry: Polygon, 15542 vertices
  ▼ properties: Object (10 properties)
    ADM0_CODE: 46
    ADM0_NAME: Canada
    ADM1_CODE: 825
    ADM1_NAME: Alberta
    DISP_AREA: NO
    EXPI_YEAR: 3000
    STATUS: Member State
    STR1_YEAR: 1000
    Shape_Area: 93.6215278576
    Shape_Leng: 44.593424804

▼ 1: Feature 00010000000000000009 (GeometryCollection, 10 properties)
  type: Feature
  id: 00010000000000000009
  ▸ geometry: GeometryCollection
  ▼ properties: Object (10 properties)
    ADM0_CODE: 46
    ADM0_NAME: Canada
    ADM1_CODE: 826
    ADM1_NAME: British Columbia / Colombie-Britannique
    DISP_AREA: NO
    EXPI_YEAR: 3000
    STATUS: Member State
    STR1_YEAR: 1000
    Shape_Area: 132.460685071
    Shape_Leng: 507.568761648

```

British Columbia / Colombie-Britannique

Climate (seasonal factor)

- ~~Precipitation, evapotranspiration, drought severity, soil moisture, max&min temperature, wind speed~~

~~[TerraClimate: Monthly Climate and Climatic Water Balance for Global Terrestrial Surfaces, University of Idaho | Earth Engine Data Catalog | Google for Developers](#)~~

Vegetation (seasonal factor)

NDVI

~~[MOD13Q1.061 Terra Vegetation Indices 16-Day Global 250m | Earth Engine Data Catalog | Google for Developers](#)~~

Topology

~~Topography information unlikely changes so the maps below can still be used.~~

- ~~Landform~~

~~[Global ALOS Landforms | Earth Engine Data Catalog | Google for Developers](#)~~

~~Is image, not image collection?~~

~~(2006-01-24T00:00:00Z-2011-05-13T00:00:00)~~

~~●—Elevation~~

[Canadian Digital Elevation Model](#) | [Earth Engine Data Catalog](#) | [Google for Developers](#)

Society

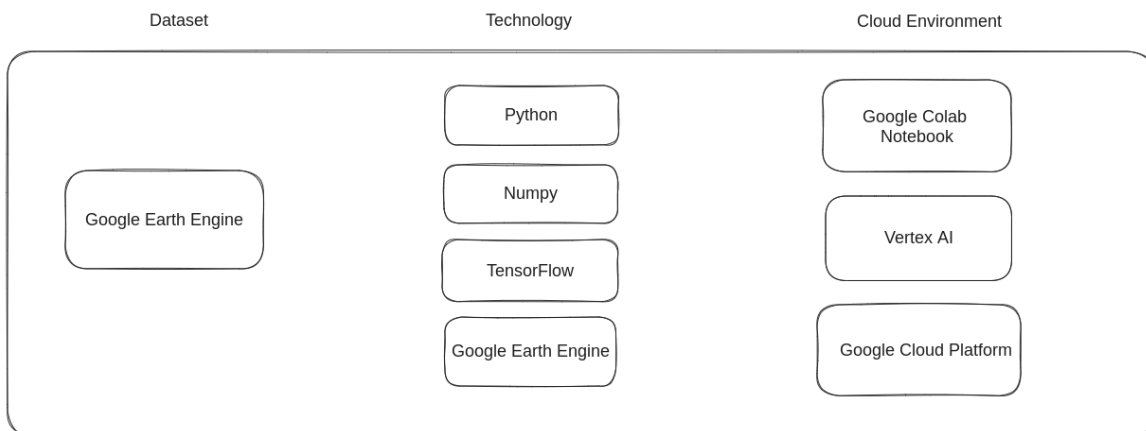
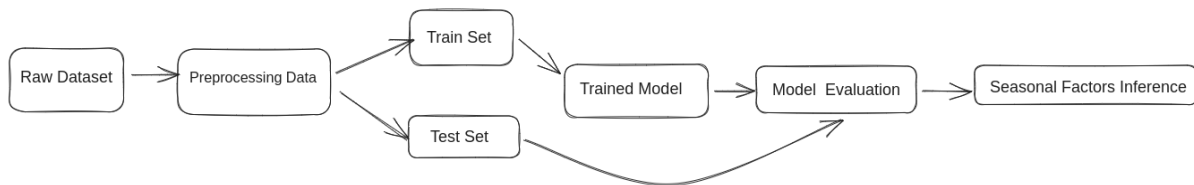
- Population

[LandScan Population Data Global 1km](#) | [Earth Engine Data Catalog](#) | [Google for Developers](#)

This needs preprocessing to measure the population within certain radius centered sample point. See [Calculate distance to target pixels in Google Earth Engine - Geographic Information Systems Stack Exchange](#)

Drawing

https://excalidraw.com/#room=99fc1b98aaa8f3e69f37.yfJCLZSI_2QFB5_whZGXUw



Problem statement

-flow chart

https://excalidraw.com/#room=99fc1b98aaa8f3e69f37.yfJCLZSI_2QFB5_whZGXUw

2. Problem Statement:

- a. Clearly define the problem statement that your research project focuses on. Describe the specific challenge or question you aim to tackle.
- b. Explain why this problem is relevant and why it is worth investigating.
- c. Consider the potential impact and benefits of solving the problem.

III. PROBLEM STATEMENT

A. Data Collection

B. Preprocessing Pipeline

C. Metrics

Each of us choose one from above section?

1. Introduction:

- a. Provide an engaging and concise introduction that presents the context and significance of the research project.
- b. Clearly state the problem that your project aims to address and its importance in the field of machine learning.
- c. Highlight the objectives and goals of your research.

2. Problem Statement:

- a. Clearly define the problem statement that your research project focuses on. Describe the specific challenge or question you aim to tackle.
- b. Explain why this problem is relevant and why it is worth investigating.
- c. Consider the potential impact and benefits of solving the problem.

Research Gap

Keywords: fire prediction "further research" OR "future work", lstm prediction "earth engine"

Justin:

-different training techniques:

loss function: relu, elu, selu, ...

techniques: [gradient clipping implementation](#)

models: CNN vs RNN

-use Canada instead of Australia <https://www.mdpi.com/2072-4292/13/1/10>

Research question 1: What are the main characteristics of the last decade's Australian wildfires from freely available satellite data?

It is essential to illustrate the exploratory analysis of historical fire events. The analysis of Australian wildfires disclosed that both 2011 and 2012 were extreme years in terms of the amount of fire activity within the study period of 2001–2019. However, the most active fires during December and January months from the last 10 years occurred in 2019 and 2020. Additionally, Australia is becoming a warmer place based on ERA5 data, which is likely due to global warming. Thus, if no mitigation and preparedness actions are undertaken, Australia will witness more wildfires and more severe wildfires in the **future**.

Research question 2: Which ML algorithm outperforms other existing models available in GEE for prediction of **future** fire occurrences?

The study compared the ML classifiers available in the GEE platform, i.e., CART, NB, and RF, and showed that the RF model outperformed the remaining ones with an overall accuracy of 96%.

Research question 3: To what extent are the various causal factors associated with the fire locations?

Using the best performing model i.e., the RF model, allowed the application of variable importance analysis. The results of variable importance analysis showed that the most important variables are soil moisture, temperature, and drought, which is in line with other studies [49,50] where these factors were also found highly important.

Forest Fire Prediction Using Heterogeneous Data Sources and Machine Learning Methods

gap:

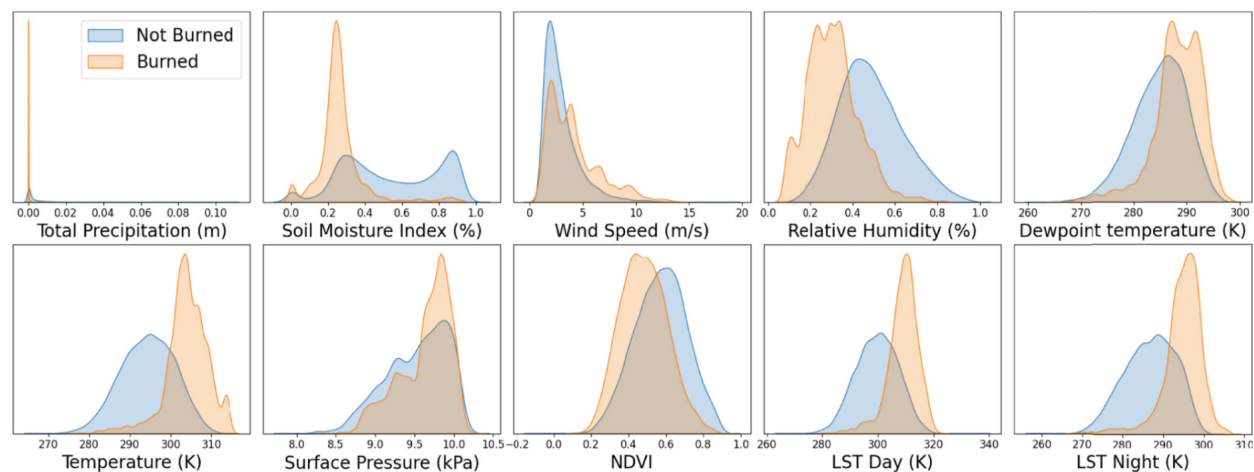
- expand the dataset: e.g. soil temperature
- LSTM
- different undersampling and oversampling methods
- data at higher resolutions

Time Series Forest Fire Prediction Based on Improved Transformer

-Improving the model's interpretability is an important direction for future work

Rice Yield Prediction in Hubei Province Based on Deep Learning and the Effect of Spatial Heterogeneity

- convLSTM
- CNN-LSTM
- preprocessing to histogram(categorical data)



source: Mia <https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2022GL099368>

Mia:

1. Count red spectrum as indicator of mountain pine beetle and add to our predictors, just focus on summer since healthy trees are green in summer while unhealthy ones are red.
[Gaps in wildfire science leave Canadian researchers fighting blind against growing risks - The Globe and Mail](#)

Costs to suppress fires have jumped roughly \$120-million a decade since the 1970s and now approach \$1-billion or more every year, posing a major challenge for cash-strapped governments at all levels. But the primary tool used by provincial wildfire agencies and crews to predict and respond to daily threats, the Canadian Forest Fire Danger Rating System, has not been updated in several decades.

That means it does not account for multiyear droughts or the spread of the mountain pine beetle, an infestation fuelled by rising temperatures that has transformed vast stretches of B.C. and Alberta into a tinderbox.

2. Seasonal predictor relevance comparison in Canada (or western Canada: BC & Alberta)
Canada official Fire Weather Index System:
[Fire Weather Index System - Province of British Columbia](#)
[Fire Weather Indices Wiki | Buildup index](#)
How are these indices calculated:
<https://wikifire.wsl.ch/tiki-index8720.html?page=Buildup+index>

Literature Review

3. Literature Review:

- Conduct a comprehensive literature review to gain an understanding of existing research and knowledge related to your problem statement.
- Summarize and analyze relevant research papers, articles, and other scholarly sources.
- Identify key theories, methodologies, and approaches used in previous work related to your research problem.
- Discuss the limitations or gaps in the existing literature that your research aims to address.

Keywords:

Semantic Segmentation

Wildfire prediction/probability/susceptibility

Solving imbalance:

[-How to Improve Class Imbalance using Class Weights in Machine Learning?.](#)
[Classification on imbalanced data | TensorFlow Core](#)

Using class weights does not increase in any way the amount of information of your dataset. In the end, using class weights is more or less equivalent to changing the output bias or to changing the threshold.

-With class weights the accuracy and precision are lower because there are more false positives, but conversely the recall and AUC are higher because the model also found more true positives.

-

Since wildfires represent only a small area of the total image pixels, we use a **weighted cross-entropy loss** to overcome the **class imbalance**. Uncertain labels are ignored in the loss and performance calculations. The models are trained over 100 epochs, with Adam optimizer, on TPUs, using

For machine learning, we include all tiles with active fires, where fire pixels that are separated by more than 10 km are considered to belong to a different fire. **We also sample twice as many tiles without fire**. We split the data between training, evaluating, and testing, by randomly separating all


Satellite image segmentation:

[A brief introduction to satellite image segmentation with neural networks | by Robin Cole | Medium](#)

<https://github.com/SatelliteVu/SatelliteVu-AWS-Disaster-Response-Hackathon/tree/main>

Mia:

1. [Hybrid artificial intelligence models based on a neuro-fuzzy system and metaheuristic optimization algorithms for spatial prediction of wildfire probability - ScienceDirect](#)
2. [Deep neural networks for global wildfire susceptibility modelling - ScienceDirect](#)
3. [Modeling spatial patterns of fire occurrence in Mediterranean Europe using Multiple Regression and Random Forest - ScienceDirect](#)
4. [Spatial and temporal patterns of wildfire ignitions in Canada from 1980 to 2006](#)
5. [Data-Driven Wildfire Risk Prediction in Northern California](#)
6. [Defining Wildfire Susceptibility Maps in Italy for Understanding Seasonal Wildfire Regimes at the National Level](#)
7. [Wildfire Danger Prediction and Understanding With Deep Learning - Kondylatos - 2022 - Geophysical Research Letters - Wiley Online Library](#)

Details of paper:  Capstone-LiteratureReview-Details

Justin:

[A Google Earth Engine Approach for Wildfire Susceptibility Prediction Fusion with Remote Sensing Data of Different Spatial Resolutions](#)

- model: learning algorithms such as artificial neural network (ANN), support vector machines (SVM), and random forest (RF) to train and test the models. This research shows that RF is slightly better than SVM. They also utilize the Dempster-Shafer Theory (DST)
- input: The input data for this study includes remote sensing data at different spatial resolutions, such as Landsat 8, Sentinel-2, ALOS, and SRTM datasets.
- output: The output data of this study are the wildfire susceptibility prediction (WSP) maps. These maps indicate the areas at risk of wildfires in the Guilan Province, Iran.
- metrics: AUC
- imbalance: Synthetic Minority Over-sampling Technique (SMOTE) algorithm. SMOTE is used to generate synthetic samples of the minority class (i.e., wildfire locations) to balance the class distribution and improve the performance of the machine learning models.
- future work: using spatially explicit polygons of burned areas as inventory data instead of commonly available points representing fire locations.

[Deep Learning Models for Predicting Wildfires from Historical Remote-Sensing Data](#)

(quite **useful**, GEE+segmentation with details) cite 12

- model: a convolutional autoencoder, a residual U-Net, a convolutional autoencoder with a convolutional LSTM, and a residual U-Net with a convolutional LSTM
- input: terrain, vegetation, and weather
- output: fire occurrence
- imbalance: 1. people also sample twice as many tiles without fire, 2. use weighted cross-entropy loss. This paper also discovered that accuracy is not a good metric because of class imbalance problems.
- metrics: AUC
- future work: Another future step is to expand these experiments beyond the United States to a global scale.

[Forest fire susceptibility assessment using google earth engine in Gangwon-do, Republic of Korea](#)

- model: classification and regression trees (CART), boosted regression trees (BRTs) and random forest (RF)
- input: distance to urban areas, rainfall amount, annual average temperature, drain density, normalized difference vegetation index (NDVI), topographic wetness index, aspect, slope, distance to rivers, distance to roads, and elevation
- output: forest fire susceptibility maps (FFSMs)
- metric: ROC and AUC
- imbalance: NA
- future work: In future studies, we will consider interrelating the fires with other disasters while maintaining the efficiency and high-accuracy of the forest fire analysis. We intend to separate the effects of human activities and interference on forest fires and evaluate the influencing factors required for the preparation of management plans.

[An alternative approach for mapping burn scars using Landsat imagery, Google Earth Engine, and Deep Learning in the Brazilian Savanna](#)

- Model: DNN
- Input: NDVI (Normalized Difference Vegetation Index), NBR (Normalized Burn Ratio), and delta NBR (Difference Normalized Burned Index)
- Output: Landsat spectral bands were used for the classification model: red (RED - 0.65 μm), near-infrared (NIR - 0.86 μm), and shortwave infrared (SWIR 1 - 1.6 μm and SWIR 2 - 2.2 μm)
- class imba: NA
- metrics: confusion matrix
- future work: /

[Bayesian prediction of wildfire event probability using normalized difference vegetation index data from an Australian forest](#)

- Model: Bayesian hierarchical logistic regression models with noninformative independent priors
- Input: NDVI, humidity, highest temperature, mean wind speed
- output: probability
- metrics: accuracy, specificity, and F1-scores
- handle imba: NA

[Wildfire Segmentation Using Deep Vision Transformers](#)

- Model: U-Net, U2-Net, and EfficientSeg. TransUNet
- input: rgb image
- output: binary mask that defines the segmented fire area in the input images
- metrics: F1-score and inference time
- cls imba: NA
- future work: the Bayesian models that are robust against a highly unbalanced distribution of zeros and ones in the dependent variable can be investigated to further improve the models considered in this study.

Vision Transformers for Remote Sensing Image Classification

- model: Vision Transformer
- input: a sequence of image patches. Each image patch represents a portion of the original image
- output: predicted class label for the input image
- metrics: classification accuracy

Huizi:

1. [Preliminary Results from a Wildfire Detection System Using Deep Learning on Remote Camera Images](#) (Cited by 69)
2. [A review of machine learning applications in wildfire science and management](#) (Cited by 335)
3. [A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms](#) (Cited by 72)
4. [Spatial Prediction of Wildfire Susceptibility Using Field Survey GPS Data and Machine Learning Approaches](#) (Cited by 105)
5. [Deep Learning and Transformer Approaches for UAV-Based Wildfire Detection and Segmentation](#) (Cited by 40)
6. [Entropy-Functional-Based Online Adaptive Decision Fusion Framework With Application to Wildfire Detection in Video](#) (Cited by 106)

1. [Preliminary Results from a Wildfire Detection System Using Deep Learning on Remote Camera Images](#) (Cited by 69)

Problem Statement:

The article addresses the challenge of detecting wildfires in their early stages. While there have been developments in networks of cameras designed to search for wildland fire signatures (like HPWREN cameras and the ALERT Wildfire camera), these cameras mainly monitor fires reported by other methods. There is a need for a system that can detect smoke from fires almost immediately after ignition.

Proposed Solution:

The authors have developed a prototype system that can detect smoke from fires usually within 15 minutes of ignition, averaging less than one false positive per day per camera. This system:

- Utilizes machine learning-based image recognition software.
- Employs a cloud-based workflow that can scan hundreds of cameras every minute.
- Operates around the clock in Southern California.
- Has detected some fires earlier than traditional methods, such as people reporting to emergency agencies or satellite detection from GOES satellites.

The authors believe that ground-based cameras alone won't be able to detect every wildfire, so they are aiming to build a system that combines the strengths of terrestrial camera-based detection with satellite-based detection.

Methods & Models:

- Machine Learning Model: The research uses the InceptionV3 image recognition model for detecting fires.
- Training the Model: They trained the InceptionV3 model on their dataset of images from cameras. They also explored the benefits of transfer learning by comparing fine-tuning a pretrained model vs. full training from scratch.

- Handling False Positives: The team implemented an algorithm to adjust the smoke threshold above 0.5 based on the maximum probability of smoke over several segments.

Preliminary Results:

- The system was able to detect smoke from fires within about 15 minutes of ignition while averaging less than one false positive per day per camera.
- The system has outperformed some commercial systems and has detected fires earlier than traditional methods like emergency calls or satellite detection from GOES satellites.
- While analyzing results, they noticed patterns in false positives, which led them to adjust their algorithms to reduce such occurrences.
- With real fires, the detection system continuously reported positive results until the fire was large enough to be detected by other means.
- They observed improvements in the model's output with every training iteration.

2. [review of machine learning applications in wildfire science and management](#) (Cited by 335)

Problem Statement:

The article addresses the application of artificial intelligence (AI) in wildfire science and management. Since the 1990s, AI methods like neural networks and expert systems have been applied to wildfire-related problems. However, with the rise of machine learning (ML) in the environmental sciences, there's a need to review ML's application in wildfire science, as well as to identify opportunities and challenges in its deployment.

Purpose:

The primary aim is to improve awareness of ML methods among wildfire researchers and managers. Additionally, the article seeks to illustrate the diverse range of problems in wildfire science that ML data scientists can address.

Key Factors & Models:

1. Domains of Application: The article categorizes ML applications in wildfire science into six problem domains:
 - Fuels characterization, fire detection, and mapping
 - Fire weather and climate change
 - Fire occurrence, susceptibility, and risk
 - Fire behavior prediction
 - Fire effects
 - Fire management
2. ML Methods Used: The most frequently used ML methods in these domains include:
 - Random forests
 - MaxEnt (Maximum Entropy)
 - Artificial neural networks

- Decision trees
 - Support vector machines
 - Genetic algorithms
3. Advantages and Limitations: The article discusses the benefits and limitations of various ML approaches, considering factors like data size, computational requirements, generalizability, and interpretability.

Preliminary Results:

1. Up to the end of 2019, 300 relevant publications on this topic were identified.
2. There exist opportunities to apply more current ML methods, including deep learning and agent-based learning, in wildfire sciences, especially for very large multivariate datasets.
3. While ML models can learn on their own, expertise in wildfire science is crucial to ensure realistic modeling of fire processes across multiple scales.
4. Some ML methods, like deep learning, have interpretability challenges, which may pose barriers in practical applications.

3. [A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms](#) (Cited by 72)

Problem Statement:

The article addresses the critical issue of wildfires, one of the most significant natural disasters threatening wildlands and forest resources. Traditional firefighting systems, which rely on ground crew inspection, have limitations and can expose firefighters to danger. As a result, there's a need for more efficient and safer wildfire detection methods.

Purpose:

The paper focuses on the application of unmanned aerial vehicles (UAVs) for early wildfire detection. Remote sensing technologies, especially those based on UAVs, have become highly sought-after strategies to combat wildfires. They offer the advantage of detecting forest fires in their early stages before they become uncontrollable. The main objective is to explore the use of deep learning algorithms for autonomous wildfire early detection from UAV-based visual data.

Methods & Models:

- Deep Learning and Computer Vision: The main focus is on deep learning algorithms combined with computer vision techniques for processing and analyzing visual data from UAVs.
- Convolutional Neural Networks (CNNs): CNNs, a type of deep learning model, seem to play a crucial role in the detection algorithms, given their prowess in image and video analysis.
- Sensor Fusion: The integration of multiple data and knowledge sources for better inference and decision-making, likely combining the data from various sensors on the UAVs.
- Remote Sensing-based Methods: Methods that use sensors from a distance, such as those mounted on watchtowers or UAVs, to detect fires or smoke.

Preliminary Results:

- **Deep Learning Performance:** The article likely discusses the performance of deep learning models in detecting wildfires from UAV-based visual data. The use of Convolutional Neural Networks (CNNs) has shown significant results in this domain.
- **Sensor Fusion:** Combining data from various sensors has likely enhanced the detection capabilities, allowing for a more accurate and comprehensive assessment of potential wildfires.
- **Comparison with Traditional Methods:** There's a comparison of the proposed methods with traditional remote sensing techniques, highlighting the benefits and potential improvements of the newer methods.

The research showcases the advancements in UAV technology combined with deep learning algorithms for early wildfire detection, emphasizing the potential of these methods over traditional techniques. The results underline the efficacy and potential of these models in real-world scenarios, providing a promising direction for future research and application in the field of wildfire detection and management.

4. [Spatial Prediction of Wildfire Susceptibility Using Field Survey GPS Data and Machine Learning Approaches](#) (Cited by 105)

Problem Statement:

Forests are ecosystems most at risk due to the consequences of climate change. One of the primary threats to forests is wildfires. The article addresses the challenge of predicting wildfire susceptibility, especially given the increasing availability of remotely sensed data, which can monitor the precise locations of wildfires reliably.

Proposed Solution:

The researchers created a wildfire data inventory by integrating global positioning system (GPS) polygons with data collected from the moderate resolution imaging spectroradiometer (MODIS) thermal anomalies product for Amol County in northern Iran from 2012 to 2017. This GPS polygon dataset was gathered through extensive field surveys. This integrated dataset, combined with several conditioning factors, was then used to evaluate machine learning (ML) approaches for spatial prediction of wildfire susceptibility.

Key Factors:

- **Data Sources:** Integration of GPS polygons and MODIS thermal anomalies product.
- **Conditioning Factors:** Sixteen conditioning factors were considered, including - topographic, meteorological, vegetation, anthropological, and hydrological factors.
- **Machine Learning:** The study leverages various machine learning approaches for spatial prediction.

Preliminary Results:

From the initial excerpt, specific results of the study have not been detailed. However, the context suggests that the article likely presents:

- Evaluation results of the proposed machine learning methods in predicting wildfire susceptibility.
- A comparison with existing methods or baseline models to demonstrate the efficacy of the proposed approach.
- Insights into the most influential conditioning factors and their role in wildfire susceptibility prediction.

In summary, the research article introduces an approach that integrates field survey GPS data and remotely sensed data to predict wildfire susceptibility using machine learning. The study leverages multiple conditioning factors to enhance prediction accuracy and provides insights into the spatial prediction of wildfire risks in forested areas.

5. [Deep Learning and Transformer Approaches for UAV-Based Wildfire Detection and Segmentation](#) (Cited by 40)

Problem Statement:

Wildfires are a global natural disaster causing significant economic damage and loss of lives. With predictions indicating an increase in wildfires in the coming years, primarily due to climate change, early detection and prediction of fire spread become crucial. While numerous systems have been developed for fire detection, recent approaches have leveraged Unmanned Aerial Vehicles (UAVs) due to their flexibility, cost-effectiveness, and ability to cover vast areas day or night. Despite their advantages, UAVs face challenges such as small fire size detection, background complexity, and image degradation.

Proposed Solution:

The authors focus on adapting and optimizing Deep Learning methods to detect wildfires at an early stage. They introduce a novel decision fusion strategy that uses transformer-based methods for wildfire segmentation and detection from UAV-based imagery.

Key Aspects:

- Deep Learning: Leveraging advanced machine learning techniques, particularly deep learning, to process and analyze UAV imagery for wildfire detection.
- Transformer Approaches: Incorporating transformer architectures, which have recently shown remarkable performance in various tasks, for wildfire segmentation and detection.
- UAV-based Detection: Emphasizing the benefits and challenges of using UAVs for wildfire detection, especially in complex environments.
- Decision Fusion: A strategy to combine decisions from multiple models or sources to enhance detection accuracy.

Preliminary Results:

From the initial excerpt, specific results of the study have not been detailed. However, given the context, the article likely presents:

- Evaluation results of the proposed deep learning and transformer methods in detecting and segmenting wildfires from UAV-based images.
- Comparisons with existing methods or baseline models to demonstrate the efficacy of the proposed approach.
- Potential challenges and limitations faced during the study and how the proposed methods address or mitigate them.

In summary, the research article introduces a novel approach that combines deep learning and transformer techniques for effective wildfire detection and segmentation from UAV-based imagery. The proposed methods aim to tackle the inherent challenges of UAV imagery and improve early wildfire detection capabilities.

[6. Entropy-Functional-Based Online Adaptive Decision Fusion Framework With Application to Wildfire Detection in Video](#) (Cited by 106)

Problem Statement:

The paper addresses the challenge of decision-making in image analysis and computer vision applications. Specifically, it focuses on scenarios where multiple sub-algorithms provide separate decisions, and there's a need for a compound algorithm that can effectively fuse these decisions to produce a final, accurate outcome. The application context is wildfire detection using video data.

Proposed Solution:

The authors introduce an Entropy-Functional-Based Online Adaptive Decision Fusion (EADF) framework. This framework assumes:

- The compound algorithm comprises several sub-algorithms.
- Each sub-algorithm produces its own decision, represented as a real number centered around zero, indicating its confidence level.
- These decision values are linearly combined using weights that are updated online through an active fusion method based on entropic projections.

The framework also takes into account feedback from an oracle (typically a human operator) to guide the decision fusion process. In the context of this study, the oracle is a security guard in a forest lookout tower who verifies the combined algorithm's decisions.

Key Aspects:

- Active Learning: The framework updates decision weights based on feedback, making it adaptive and improving its performance over time.
- Entropic Projections: These are used to project decision values onto convex sets describing the sub-algorithms, which helps in adjusting weights and making the final decision.
- Application to Wildfire Detection: The framework's application is showcased in the context of video-based wildfire detection.

Preliminary Results:

- The EADF framework was applied to video-based wildfire detection, with image data arriving sequentially.
- The oracle, in this case, was a security guard who provided real-time feedback.
- The paper likely presents simulation results to showcase the efficacy of the proposed decision fusion method in wildfire detection.

In summary, the research introduces a novel decision fusion method for image analysis and computer vision applications, specifically highlighting its benefits in video-based wildfire detection. The proposed framework dynamically adjusts decision weights based on feedback and the confidence levels of individual sub-algorithms, aiming to enhance accuracy and adaptability.