Today I'll be sharing some of the technical and data science information that we learned from our recent AI/ML challenge.

The OCTO works to promote emerging technology and spread understanding of technology in the agency.
During the summer of 2020, the OCTO and FAS ITC collaborated to host an online machine learning challenge.

Full details are at this website:
Let's start with the business problem wanted to solve:

The goal of this challenge is to develop an artificial intelligence (AI), or machine learning (ML) solution that will review end-user license agreements (EULA) for terms and conditions that are unacceptable to the government. On average it takes all parties involved approximately 7-14 days to review an EULA and ensure that all unacceptable terms and conditions are identified.

A EULA details the rights and restrictions which apply to the use of software or services. It can also be known as a software license agreement or acceptable use policy. As part of the acquisition process of software or services, GSA reviews the associated EULAs. This review must be completed prior to awarding new contracts or modifying existing contracts. A GSA contracting officer (CO) reviews applicable EULAs for terms and conditions that are not in accordance with Federal law and regulations. The CO may also coordinate a legal review with the Office of General Counsel if they feel it is warranted. Should EULAs contain language that would conflict with Federal law and regulations, the CO must negotiate changes to the EULA to remove the problematic language.

We are looking for a solution that will use AI and/or ML to improve this
manual process. The solution will include a user interface that GSA will use to process the documents and identify unacceptable clauses/terms in the EULAs. Watch our AI / ML challenge video to learn more about the desired solution.

This solution will decrease the time spent manually reviewing EULAs and free resources to focus on other aspects of the acquisition process. It will also improve the accuracy and consistency of the review process.
There are a variety of reasons a clause might be unacceptable to government. We provided “Attachment B” to the teams in the reference materials: https://github.com/GSA/ai-ml-challenge-2020/tree/master/reference
This shows how AI & ML relate to each other.
This is a typical machine learning flow.
This is a view of a machine learning pipeline.
https://towardsdatascience.com/architecting-a-machine-learning-pipeline-a847f094d1c7
We had instructions on challenge.gov and github. Here were the github instructions:
These were the scoring criteria. We published a scoring rubric to explain them. [https://github.com/GSA/ai-ml-challenge-2020/blob/master/reference/Al_ML%20Challenge%20Scoring%20Rubric.pdf](https://github.com/GSA/ai-ml-challenge-2020/blob/master/reference/Al_ML%20Challenge%20Scoring%20Rubric.pdf)
These were the three winners.
Welcome to EULACheck, Dev Technology’s End User License Agreement (EULA) analysis app!

You may upload a EULA in Word (.docx) or PDF (.pdf) format, or paste EULA content directly into the text box from your clipboard.

We will take a look at the first 10 clauses, and determine if the Federal Government is likely to object to the language there.

We will also show you clauses that closely match yours which are known to be unacceptable and/or acceptable to the Federal Government.

All content on this site is informational - it should not be construed as legal advice.

Finally, you can provide your feedback by telling us if our analysis was correct or not.

*The application works best on Chrome or Firefox

'...' the Customer does not make any admissions (save where required by court order or governmental regulations, and where the Customer is required under the terms of such order or regulations not to first consult with the Company) which may be prejudicial to the defense or settlement of any Claim without the Company’s approval (not to be unreasonably withheld or delayed)."
Requests. Company will notify Customer before Customer exceeds the Tile Request Use Limit indicated on the Order Form. If Customer exceeds its Tile Request Use Limits during the License Term, Company will invoice Customer for overages on written notice (which may be by email). If, after 30 days from the date of that written notice, Customer continues to exceed its Tile Request Use Limit, Company may stop providing the Service to the Customer initiate a claim with the Contracting Officer under the Contract Disputes Act.

<table>
<thead>
<tr>
<th>Clause</th>
<th>Acceptable?</th>
<th>Feedback</th>
<th>Closest Acceptable Match</th>
<th>Closest Unacceptable Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requests. Company will notify Customer before Customer exceeds the Tile Request Use Limit indicated on the Order Form. If Customer exceeds its Tile Request Use Limits during the License Term, Company will invoice Customer for overages on written notice (which may be by email). If, after 30 days from the date of that written notice, Customer continues to exceed its Tile Request Use Limit, Company may stop providing the Service to the Customer initiate a claim with the Contracting Officer under the Contract Disputes Act.</td>
<td>Acceptable</td>
<td>Acceptable</td>
<td>Term of this Addendum. This Addendum will commence on the Addendum Effective Date and continue for a period of twelve months (&quot;Initial Addendum Term&quot;). Upon the effective date of termination of this Addendum in accordance with the Contract Disputes Act, Client's access to the Hosted Service provided pursuant to this Addendum (and all Services granted under this Addendum) will cease and COMPANY will delete all backed-up Client Data from the Hosting Infrastructure within 30 days of termination of this Agreement.</td>
<td>Company warrants that the Service will, for a period of sixty (60) days from the date of your receipt, perform substantially in accordance with Service written materials accompanying it.</td>
</tr>
</tbody>
</table>
Upload the document you want analyzed

Select file... Browse

Analyze
### Showing analysis for sample_eula_1.docx

<table>
<thead>
<tr>
<th>Clause</th>
<th>Recommended</th>
<th>Confidence</th>
<th>Reviewer’s Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTER SERVICES SUBSCRIPTION AGREEMENT</td>
<td>Accept</td>
<td>94.32</td>
<td>Agree</td>
</tr>
<tr>
<td>This Master Services Subscription Agreement (the “Agreement”) sets forth the terms and conditions governing COMPANY provision to Client of a cloud-based asset management and decision support system.</td>
<td>Accept</td>
<td>85.5</td>
<td>Agree</td>
</tr>
<tr>
<td>This Agreement, including the Order Form attached to it, as well as any Order Forms and Statements of Work entered into by the parties from time to time, the underlying AGENCY Schedule Contract, and Schedule Pricelist, together constitute the entire agreement of the parties and supersedes any prior and contemporaneous oral or written understanding as to the parties’ relationship and the subject matter hereof. In the event of any conflict or contradiction among the foregoing documents, the documents will control in the order listed in Contract Clause 552.212-4(a). This Agreement may only be amended in a writing signed by both parties.</td>
<td>Reject</td>
<td>39.03</td>
<td>Agree</td>
</tr>
<tr>
<td>This Agreement may be executed in two or more counterparts, each of which will be deemed an original for all purposes, and together will constitute one and the same document. Once signed, both parties agree that any</td>
<td>Accept</td>
<td>89.71</td>
<td>Agree</td>
</tr>
</tbody>
</table>
Welcome!
Please enter your End-User License Agreement (EULA):
<table>
<thead>
<tr>
<th>Clause Text</th>
<th>MF Score (%)</th>
<th>SRV Score (%)</th>
<th>SLA Score (%)</th>
<th>Acceptability (Avg Score)</th>
<th>User Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Warranty</td>
<td>100</td>
<td>99</td>
<td>100</td>
<td>Yes (100%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>2. Support</td>
<td>92</td>
<td>100</td>
<td>100</td>
<td>Yes (92%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>3. Compliance</td>
<td>97</td>
<td>100</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>4. Termination</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>Yes (100%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>5. Data Rights</td>
<td>98</td>
<td>100</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>6. Limitation of Liability</td>
<td>93</td>
<td>99</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>7. Indemnification and Limitation of Liability</td>
<td>99</td>
<td>100</td>
<td>100</td>
<td>Yes (100%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>8. Governing Law</td>
<td>98</td>
<td>87</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>9. Entire Agreement</td>
<td>100</td>
<td>99</td>
<td>100</td>
<td>Yes (100%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>10. Governing Law</td>
<td>94</td>
<td>74</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
<tr>
<td>11. Entire Agreement</td>
<td>97</td>
<td>87</td>
<td>100</td>
<td>Yes (99%)</td>
<td>Open - Disable</td>
</tr>
</tbody>
</table>
How Do These Apps Work?
We found a lot of similarities between the solutions we received.

This is a big-picture view of the typical architecture that teams used.
How Did The Teams Work?
I am going to demonstrate the process followed by one of the teams.
The team used Jupyter Notebook and Python for their data science work.
This shows their finished architecture.
This is how the team described their process for deciding on the machine learning libraries to use:

- As a first step, our team used traditional algorithms to approach the problem.
- We started with **Random Forest**, which is a popular machine learning algorithm that is widely used in classification tasks.
- **Random Forest** is simple and intuitive in nature. It does not require hyperparameter tuning and usually does not overfit to the dataset with an increase in the number of decision trees within the model. We achieved an F1 score of 27.5% with it.
- As the next step, we implemented the **XGBoost** algorithm. XGBoost is a tree-based algorithm (like Random Forest) but uses the technique of boosting. Boosting is an error-correction algorithm which gives a higher emphasis on data-points which are misclassified. Unlike random forests, the decision trees are created iteratively, where at each step, the tree puts more emphasis on the misclassified points, so as to reduce the overall error. We therefore used XGBoost as the natural next step to Random Forest. On testing however, the F1 score achieved through XGBoost was lower than Random Forest. This was potentially due to a high False Negative, resulting in a low Recall value. This means that the model was classifying most clauses
(even the ones that were labelled unacceptable) as acceptable.

After running several experiments trying to improve the accuracy of traditional models, we realized that more advanced, deep learning based models could potentially help us increase accuracies. We therefore started with the simplest form of sequence models that are used on textual data: Recurrent Neural Networks (RNN). As a starting point we used a Long Short-Term Memory (LSTM) which attempts to resolve the vanishing gradient, a known obstacle for RNNs. LSTMs however capture the flow of information in one direction (left to right in case of sentences). Bidirectional LSTMs capture the flow of information from both left to right and right to left. This serves as an advantage as the model can learn from the future and the past information at a given point in the sentence. To add interpretability to our results, we added an attention layer that provides the importance of words in the decision making process of the model. The attention layer weighs those words differently providing more emphasis on words that have stronger relationships. Experiments run by researchers and practitioners have shown that Gated Recurrent Units (GRUs) train faster and provide a proportional (or sometimes better as in our case) accuracies to that of LSTMs. We therefore replaced the LSTM units with GRUs. The GRU unit does not have a forget gate (unlike the LSTM) and has fewer parameters to train on.

Arguably the most sophisticated classification model, as of now, is the XLNet which builds upon the transformer architecture which incorporates encoding and decoding layers in addition to sinusoidal position encoding of words in a sentence. It also incorporates permutations of words to learn more complex relationships between words. It has been pre-trained on a large corpus of data from Wikipedia, BookCorpus, etc. In our experiments, it did improve the validation accuracies.

To provide an ensemble solution, we combined our best performing models: XLNet and Bi-GRU with attention. The final classification decision made by this ensemble was the average of the probabilities coming out of these models. We had lively discussions about whether we should just keep XLNet or do a weighted combination of XLNet with Bi-GRUs. The result was that the argument for robustness in the estimate ensembling these two complex algorithms won the argument of the day. In addition, in our final solution, we kept the Random Forest score as a sanity check output in our display. We believe this model is
• the easiest to explain to lay people. However this score was not used in the average.
From the team:
It comprises two major components: Words to Numbers and Classification. Arrows show which words to numbers algorithm used for which classification algorithm. Solid lines represent the models that were integrated into the final solution, while dashed lines represent the ones that we tested during the development process of our final solution.
Thanks!