turkcellproto

prototype – ml for radio link failure prediction, prescription, and prevention

faiz ikramulla

dhruv gaonkar

jeff anderson

aaron hoock

Aclara/Hubbell, October 2020

Approach

- Platform selection (Colaboratory + Python packages)
- Exploratory data / statistical analysis
- Visualizations
- Data cleansing / prep
- Feature engineering
- Proto modeling
- Initial ML Training / Test
- Next steps

Dataset

- 180000+ rows over entire 2019 (various seasons, locations, etc.)
- 611 radio link failures (179699 radio link passes)
- 0.34% failure rate (99.66% passing rate)
- Binary classification problem – model should be better than 99.66%!
- Some prelim statistical/correlation analysis - **Much more to do as next steps*
- Feature engineering to simplify dataset and extract useful info

Machine Learning for RF Link Prediction

- Start small – scale fast
    - Dummy variables on categorical values
    - Eliminate redundant features
    - Identify “key players” based on domain expertise: freq_band, rxlevmax, modulation type, capacity, etc.
- We started by preparing/cleaning up the data a bit
- We remove some columns for our initial modeling. **Note – we need to revisit this in next steps as we removed spatial and weather components as a starting point ONLY.*

Stats / Visualizations

```
f =df3['rlf_True'].sum()
f
```

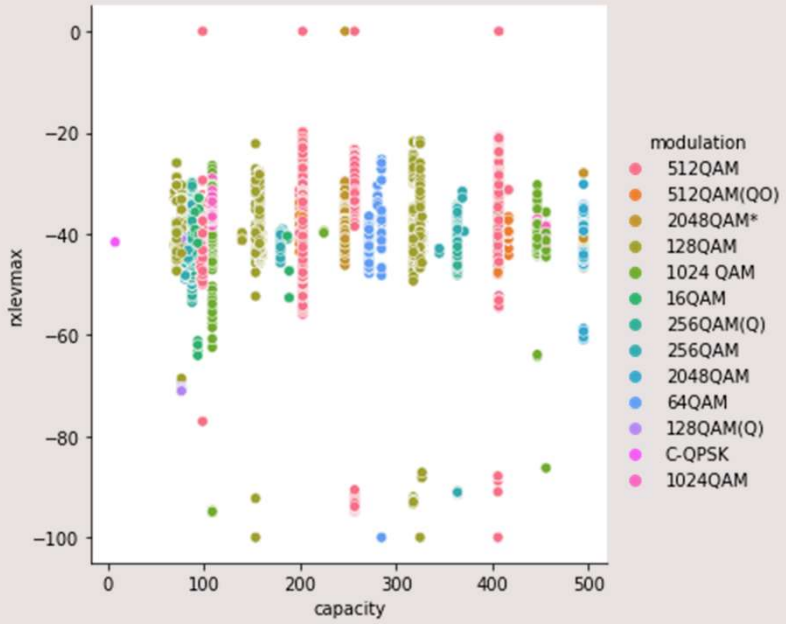
```
611
```

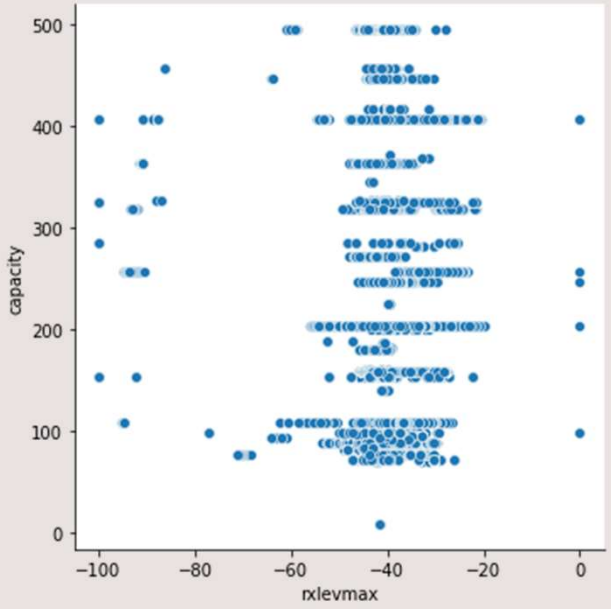
```
p = df3['rlf_False'].sum()
p
```

```
179699
```

```
f/(f+p)*100
```

```
0.33886085075702954
```





Data Preparation / Cleansing

- We dropped the following columns for our initial modeling:
- 'Unnamed: 0','site_id','site_no','type','datetime','scalibility_score','tip','mlid','mw_connection_no','neid','direction','polarization','link_length','severaly_error_second','error_second’
- We want to look at RF/network parameters first.
- Then, we plan to (as next steps) add spatial, temporal, and weather data back to enrich our data and model.

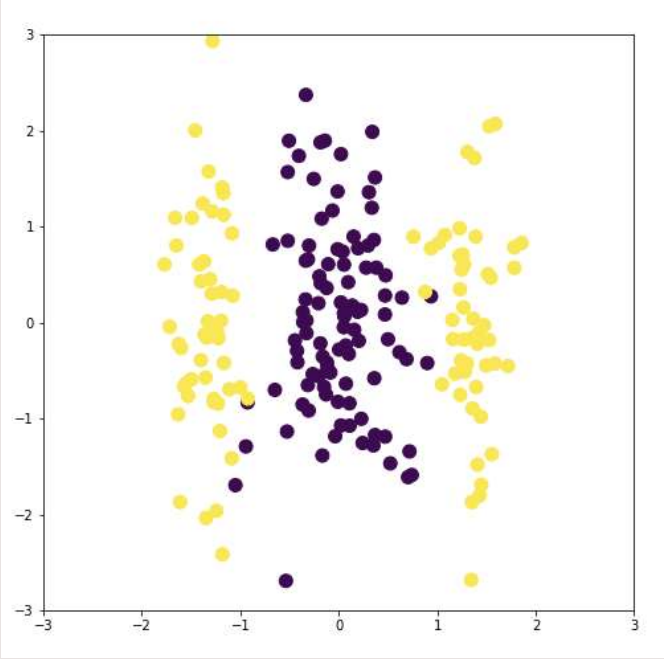
Machine Learning Modeling

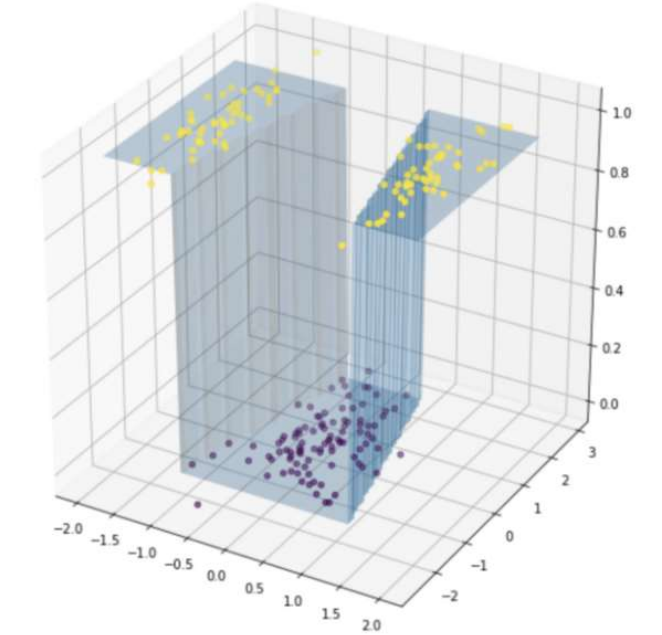
- 80/20 Train / Test Split
- Many modeling to choose from in scikitlearn
- For our number of features and desired lable (0, 1) – we choose SVM method with a default (linear) kernel. **as starting point only.*

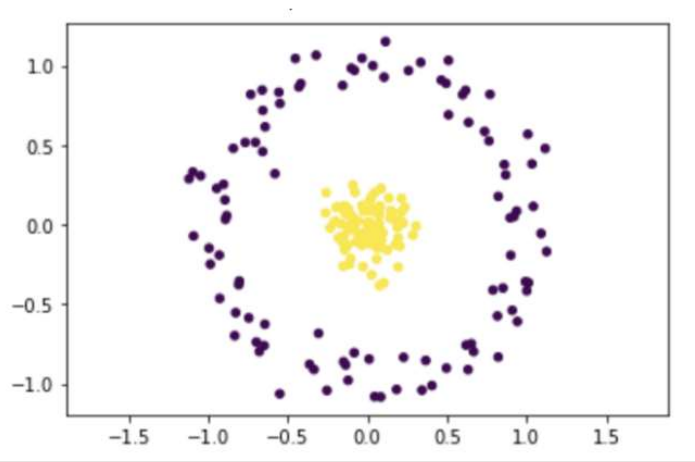
```
svc = SVC(kernel='linear', C=10.0, random_state=1)

svc.fit(X_train, y_train)
```

Machine Learning – Ex. Of SVM

https://towardsdatascience.com/animations-of-neural-networks-transforming-data-42005e8fffd9

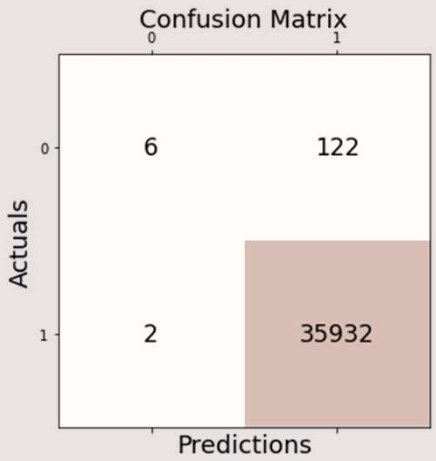






Machine Learning - Test

- About 271 seconds training time in Colaboratory GPU runtime.
- Results on test set: (f1 score = 99.8%)




```
# metrics
print('Precision: %.3f' % precision_score(y_test, y_pred))
print('Recall: %.3f' % recall_score(y_test, y_pred))
print('Accuracy: %.3f' % accuracy_score(y_test, y_pred))
print('F1 Score: %.3f' % f1_score(y_test, y_pred))
```

```
Precision: 0.997
Recall: 1.000
Accuracy: 0.997
F1 Score: 0.998
```

Machine Learning – Status Today

- Our default model has an f1 score = 99.8%, better than the target of 99.66%.
- We have not (yet) thoroughly checked for correlations/cross correlations between the features as-is.
- We did not (yet) include spatial, temporal, seasonal, weather features – only RF parametrics thus far (as a starting point)

Machine Learning – Next Steps

- Now that we have a good starting point, we plan to:

- *Refine model by checking for correlation and cross correlation of RF / network features*
- *Subset model based on modulation type, freq band*
- *Add temperature / wind as features on each subset and re-train*
- *Add geodistances and features on each subset and re-train*

Machine Learning – Proposed Final / Workflow

- “Ensemble Model” – Optimized for each network type (mod scheme and freq band) , to be predictive across all weather types and locations
- “Prescription / Prevention” – on how to predict & handle soon-to-be radio link failures (i.e. soft-handover to another band or mod scheme to maintain link based on real-time operating ML model, or other mechanism, to avoid link droppage, at-speed and safely!

References

https://vitalflux.com/accuracy-precision-recall-f1-score-python-example/

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

https://towardsdatascience.com/animations-of-neural-networks-transforming-data-42005e8fffd9

https://seaborn.pydata.org/tutorial/relational.html

https://scikit-learn.org/stable/modules/model_evaluation.html