Convergent Genomics Data Science Challenge

The data set is explored through below steps:

1. Data frame building and cleaning
2. Feature engineering
3. Modeling building
4. Model selecting
5. Feature importance analysis

1. Data frame building and cleaning:

    - The data frame was joined based on the sample ID of mrna_data, seq_data and patient_data after basic string cleaning. Redundant features in each data set were removed from the initial cleaning based on their distribution and explanatory characters.
    - The data frame is attached in the folder and named as `df.csv`. It's a data frame with 531 observations and 195 features.
    - Tested kNN, EM and removing strategies on missing value imputation. The kNN strategy is selected based on the performance of value estimation.

2. Feature engineering:

    - Categorical data `'American Joint Committee on Cancer Tumor Stage Code'` was recategorized as four main categories: `'T1', 'T2', 'T3', 'T4'` to boost model performance and enhance feature significance.
    - Checked the distribution of each feature and removed features with extreme imbalanced distribution or strong correlation.
    - The features correlation matrix indicates that numeric features mostly have low correlations.

3. Modeling building:

    - Applied the most fundamental classification models: `Logistic Regression`, `kNN`, `Random Forest` and `Gradient Boosting Decision Tree` to test model performance on testing set through patients/mrna/sequence features against patient's `'Overall Survival Status_DECEASED'`.
    - Conducted a `10-folds cross validation` to double verify models' bias and variance.

4. Model selecting:

    - Tuned each model's hyper parameters based on `10-folds validation` performance. `Logistic Regression` and `Gradient Boosting Decision Tree` generally have the best performance in terms of models' precision and recall rate.
    - Applied the parameter-tuned model on the testing set and checked the `confusion matrix` and `ROC curve`.
    - Worked on the trends between threshold and precision/recall to decide the best performing threshold of maximizing precision/recall. In this case, the model is designed to weight more on recall rate since detecting all `Positive Deceased` is more important to our findings.

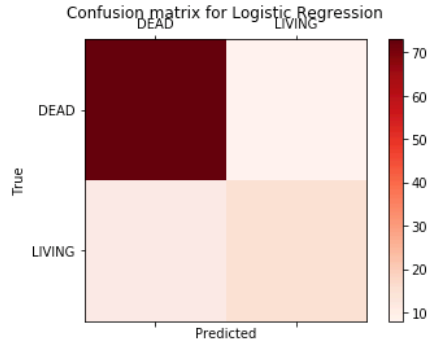
5. Feature importance analysis:

    - Conducted feature importance analysis via `L1-regularized Logistic Regression coefficient` and `Random Forest feature importance`.
    - Find the common important features and adjust the input data to re-test models.

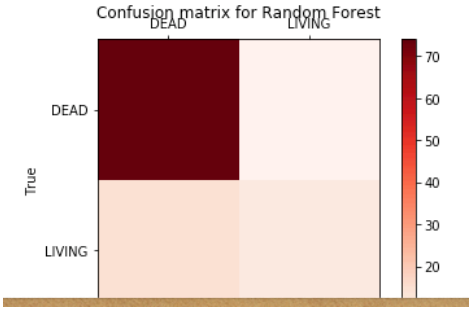
Model Performance:

After the parameter tuning, here are the scores of each model:

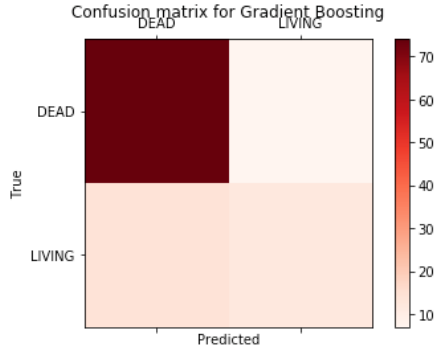
```
Logistic Regression
Accuracy is: 0.822
precision is: 0.652
recall is: 0.577
```




```
Random Forest
Accuracy is: 0.794
precision is: 0.611
recall is: 0.423
```

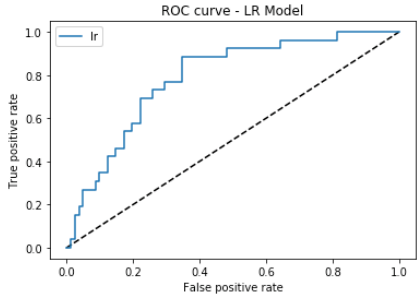



```
Gradient Boosting
Accuracy is: 0.804
precision is: 0.632
recall is: 0.462
```



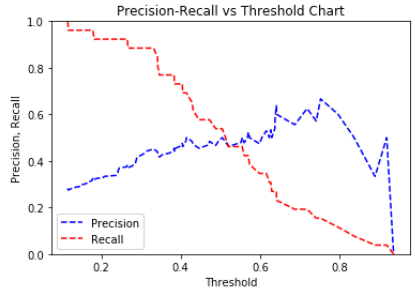
The best model is `Logistic Regression` under this training and testing case.

```
from sklearn.metrics import roc_curve
from sklearn import metrics

# Use predict_proba to get the probability results of Logistic Regression
y_pred_lr = best_LR_model.predict_proba(x_test)[:, 1]
fpr_lr, tpr_lr, _ = roc_curve(y_test, y_pred_lr)
# ROC Curve
plt.figure(1)
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr_lr, tpr_lr, label='lr')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve - LR Model')
plt.legend(loc='best')
plt.show()
```




```
: from sklearn.metrics import precision_recall_curve
precision, recall, thresholds = precision_recall_curve(y_test, y_pred_lr)
   #retrieve probability of being 1(in second column of probs_y)
pr_auc = metrics.auc(recall, precision)

plt.title("Precision-Recall vs Threshold Chart")
plt.plot(thresholds, precision[: -1], "b--", label="Precision")
plt.plot(thresholds, recall[: -1], "r--", label="Recall")
plt.ylabel("Precision, Recall")
plt.xlabel("Threshold")
plt.legend(loc="lower left")
plt.ylim([0,1])
```

```
: (0, 1)
```



From the graphs above, we optimize the threshold to get a recall-weighted model based on the recall curve derivate. The optimized threshold is around 0.3.
We take the `deceased_rate` as the threshold. The performance of the model is:

```
(y_train.sum()/len(y_train))[0]
```

```
0.35058823529411764
```

```
threshold_performance(best_LR_model, (y_train.sum()/len(y_train))[0])
```

```
LogisticRegression(C=0.05, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
Threshold is: 0.351
Accuracy is: 0.785
precision is: 0.537
recall is: 0.846
```

Here we have the final optimized model with `Recall = 0.846` and `Precision = 0.537`.

L1-regularized Logistic Regression coefficient:

```
x = df_knn.drop(['Overall Survival Status_DECEASED'],axis=1)
y = df_knn['Overall Survival Status_DECEASED']

LRmodel_l1 = best_LR_model
LRmodel_l1.fit(x, y)
LRmodel_l1.coef_[0]
print ("Logistic Regression (L1) Coefficients")
for k,v in sorted(zip(map(lambda x: round(x, 4), LRmodel_l1.coef_[0]), \
                      x.columns), key=lambda k_v:(-abs(k_v[0]),k_v[1])):
    print (v + ": " + str(k))
```

```
Logistic Regression (L1) Coefficients
Stage_T1: -0.4631
Neoplasm Histologic Grade_G2: -0.4381
Diagnosis Age: 0.4318
Neoplasm Histologic Grade_G4: 0.3915
SHISA6: -0.3118
HIST1H2AM: 0.2776
DGKK: -0.2757
PCDHB1: -0.2712
SLITRK6: 0.2439
NTRK2: -0.2378
CXADRP3: 0.2302
DPEP1: -0.2087
RLN1: -0.2078
PCP4: -0.2073
RIIAD1: -0.2054
BEX1: 0.1895
STARD6: 0.1845
Race Category_BLACK OR AFRICAN AMERICAN: -0.1799
```

Random Forest feature importance:

```
# check feature importance of random forest for feature selection
x_feature_importance = df_knn.drop(['Overall Survival Status_DECEASED'],axis=1)
y = df_knn['Overall Survival Status_DECEASED']

forest = RandomForestClassifier()
forest.fit(x_feature_importance, y)

importances = forest.feature_importances_

# Print the feature ranking
print("Feature importance ranking by Random Forest Model:")
for k,v in sorted(zip(map(lambda x: round(x, 4), importances), x_feature_importance.columns), reverse=True):
    print (v + ": " + str(k))
```

```
Feature importance ranking by Random Forest Model:
ROS1: 0.0302
Neoplasm Histologic Grade_G4: 0.0286
Diagnosis Age: 0.0282
HIST1H2AM: 0.0256
SERPINB2: 0.0245
Stage_T1: 0.0229
BNC1: 0.021
EPS8L3: 0.0199
SPON1: 0.0193
NPB: 0.0193
USH1G: 0.0191
UBE2QL1: 0.0187
BEX1: 0.0177
DGKK: 0.0173
SCG2: 0.017
NTRK2: 0.0164
LCN10: 0.0162
TCEAL2: 0.0161
```

From the data above, we can conclude that the most important features to the model prediction are `'recategorized American Joint Committee on Cancer Tumor Stage Code', 'Neoplasm Histologic Grade'` and some other mrna features like: `'ROS1', 'HIST1H2AM'` and etc.

1. *What features of the data are most important for QC/QA?*

As the conclusion shows above, the most important features are: are recategorized `'American Joint Committee on Cancer Tumor Stage Code', 'Neoplasm Histologic Grade'` and some other mrna features like: `'ROS1', 'HIST1H2AM'` and etc.

2. *Generally speaking, what are potential sources of ambiguity arising from your approach?*

The potential ambiguity mostly come from the limitation of the data set size. There are around 200 features and only 500 observations, which makes the model performance unstable and also impairs the model performance. Different approaches to process missing data and other feature engineering methods also increased the ambiguity of the final model.

3. *What other data might we collect to enhance risk quantification? What quantitative proof do you have?*

As what the current model shows, a detailed sequence data of those important mrna ('ROS1', 'HIST1H2AM' and etc.) might be helpful to boost the model performance. Other patients' health condition indices correlated with the cancer stage will be also helpful to the improve the model.

4. *Describe your approach to filing IP claims around your unique classification of risk?*

The risk I assessed is mainly the 'deceased risk'. There are other ways to do some compound metrics to assess risk levels but it has to be tested based on the significance via models.

5. *How would you communicate your findings to a clinician?*

As for the 'deceased risk', I would directly state the probability of the patients' decease based on the recall rate. I would also explain the reason based on the feature importance. I may suggest to do some further test on high-risk patients to confirm their data and may need more types of features from those high-risk patients to adjust the model.