1. Test out all 25-30 classifier models including Catboost and LogosticGAM on both overall as well as approval & denial accuracy, with onehot, target+onehot encoding, showing results in below format. Also refer [this kernel [21]](https://www.kaggle.com/ldfreeman3/a-data-science-framework-to-achieve-99-accuracy/notebook) for comparison table code.

models = pd.DataFrame({

'Model': ['Support Vector Machines', 'KNN', 'Logistic Regression',

'Random Forest', 'Naive Bayes', 'Perceptron',

'Stochastic Gradient Decent', 'Linear SVC',

'Decision Tree'],

'Score': [acc\_svc, acc\_knn, acc\_log,

acc\_random\_forest, acc\_gaussian, acc\_perceptron,

acc\_sgd, acc\_linear\_svc, acc\_decision\_tree]})

models.sort\_values(by='Score', ascending=False)

1. Logistic coefficient table view

coeff\_df = pd.DataFrame(train\_df.columns.delete(0))

coeff\_df.columns = ['Feature']

coeff\_df["Correlation"] = pd.Series(logreg.coef\_[0])

coeff\_df.sort\_values(by='Correlation', ascending=False)

1. Learn hyperparameters tuning, Classifier CV comparison and RFE feature selection and voting classifier from [this](https://www.kaggle.com/ldfreeman3/a-data-science-framework-to-achieve-99-accuracy/notebook) kernel. [This](https://www.kaggle.com/yassineghouzam/titanic-top-4-with-ensemble-modeling) kernel is also one of the best.
2. Plot multiple confusion matrix

f,ax=plt.subplots(3,3,figsize=(12,10))

y\_pred = cross\_val\_predict(svm.SVC(kernel='rbf'),X,Y,cv=10)

sns.heatmap(confusion\_matrix(Y,y\_pred),ax=ax[0,0],annot=True,fmt='2.0f')

ax[0,0].set\_title('Matrix for rbf-SVM')

y\_pred = cross\_val\_predict(svm.SVC(kernel='linear'),X,Y,cv=10)

sns.heatmap(confusion\_matrix(Y,y\_pred),ax=ax[0,1],annot=True,fmt='2.0f')

ax[0,1].set\_title('Matrix for Linear-SVM')

plt.subplots\_adjust(hspace=0.2,wspace=0.2)

plt.show()

1. Setting data type while loading the data

train = pd.read\_csv('../input/train.csv', header = 0, dtype={'Age': np.float64})

1. Seaborn aesthetics

sns.despine(offset=10, trim=**True**);

sns.set\_style("whitegrid")

sns.boxplot(data=data, palette="deep")

sns.despine(left=**True**)

1. Use t-SNE to plot data in 2d color-coded with target variable to identify if approval and denied cases are seperable or not. Use autoencoders if seperation is not present. Refer to [this](https://www.kaggle.com/shivamb/semi-supervised-classification-using-autoencoders) kernel for more information. 
2. Permutation Importance – works on any model

import eli5

from eli5.sklearn import PermutationImportance

perm = PermutationImportance(my\_model, random\_state=1).fit(val\_X, val\_y)

eli5.show\_weights(perm, feature\_names = val\_X.columns.tolist())

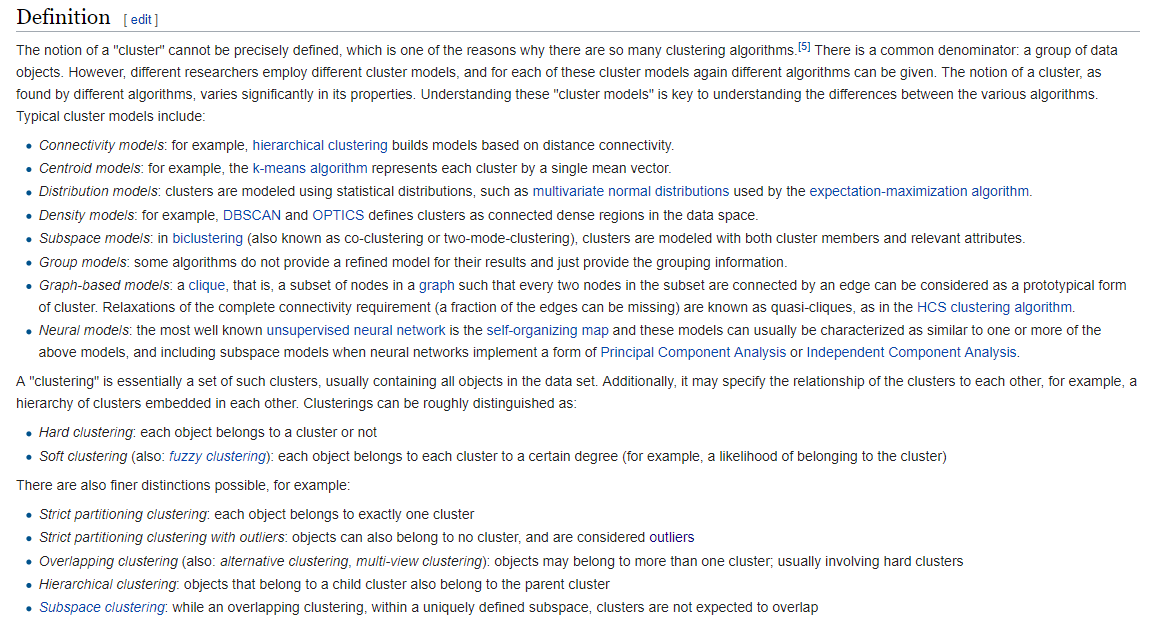
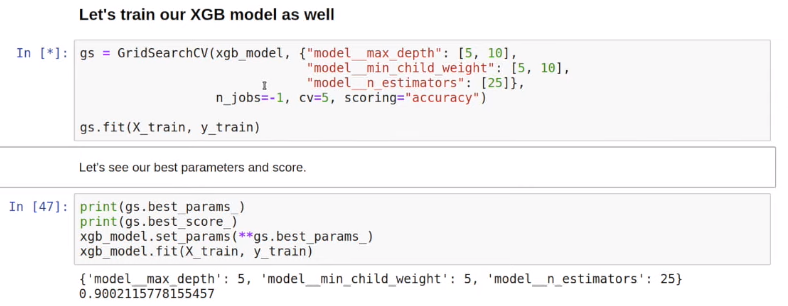
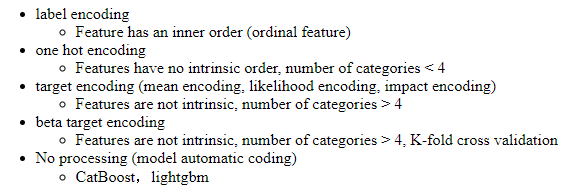
#Explain model globally

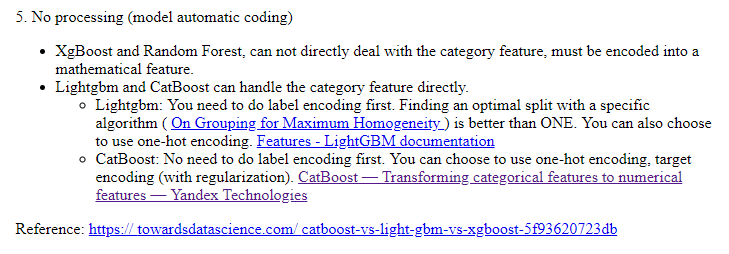
Eli5.show\_weights(model)

#Explain a single explanation

Eli5.show\_prediction(model, observation)

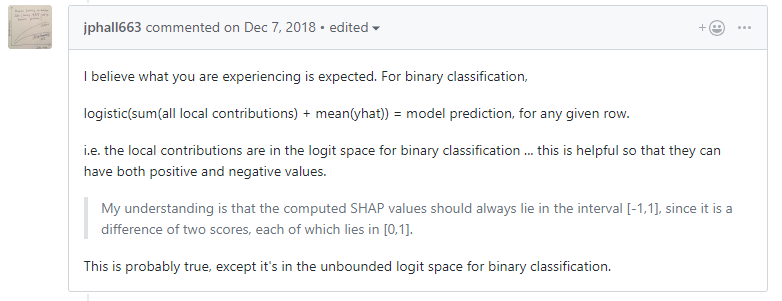
show\_prediction(clf, valid\_xs[1], vec=vec, show\_feature\_values=**True**)

1. [Dealing](https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/) with imbalanced classes
   1. Oversample using SMOTE and MSMOTE and check train, test, approval and denial accuracies
   2. Hypertuned Bagging and Boosting classifiers and check metrics
2. Bagging oob\_score = True gives the idea of test accuracy -> bag\_clf.oob\_score\_. Use warm\_start = 1 during randomizedsearchcv
3. Important features
   1. Procedure and Diagnosis code
   2. Length of stay
   3. Frequency of service
   4. Treatment type
   5. Facility provider type
   6. Patient age
4. **Is Approved/Reject right? Or should we consider Approved and Not-approved (adding all other categories), similar to that of HCSC use-case**
5. 
6. How kagglers do EDA - Understand the problem. We'll look at each variable and do a philosophical analysis about their meaning and importance for this problem
7. Correlation analysis between 2 categorical variables – Chi2 test.
   1. Null hypothesis: they are independent, Alternative hypothesis is that they are correlated in some way.
   2. Correlation between independent variables should be low (no multicollinearity) and correlation between an independent and dependent variable should be high.
   3. For categorical, we use lambda, Cramer V instead of Pearson as a metric.
   4. The Fisher’s exact test is used when you want to conduct a chi-square test, but one or more of your cells has an expected frequency of five or less. Remember that the chi-square test assumes that each cell has an expected frequency of five or more, but the Fisher’s exact test has no such assumption and can be used regardless of how small the expected frequency is.
8. Different types of distance metrics
   1. Sum of Absolute Distance
   2. Sum of Squared Distance
   3. Mean-Absolute Error
   4. Euclidean Distance
   5. Manhattan Distance
   6. Chessboard Distance
   7. Minkowski Distance
   8. Canberra Distance
   9. Cosine Distance
   10. Hamming Distance
9. Make your code PEP8 compatible. Pycodestyle. Autopep8. Pylint.
10. magic functions: %%prun, %%memit
11. WoE and IV concept: The weight of evidence (WOE) and information value (IV) provide a great framework for exploratory analysis and variable screening for binary classifiers. WOE and IV have been used extensively in the credit risk world for several decades, and the underlying theory dates back to the 1950s. However, it is still not widely used outside the credit risk world and it is a somewhat underserved area in R.
    1. WOE describes the relationship between a predictive variable and a binary target variable.
    2. IV measures the strength of that relationship.
    3. Excellent blog article: <https://multithreaded.stitchfix.com/blog/2015/08/13/weight-of-evidence>
12. Comparative analysis of different explainers
    1. Skater - <https://github.com/datascienceinc/Skater>
    2. Model-independent explanation methods - <http://lkm.fri.uni-lj.si/rmarko/papers/RobnikSikonjaKononenko08-TKDE.pdf>
    3. Interactions of subsets of feature values (<http://lkm.fri.uni-lj.si/xaigor/slo/pedagosko/dr-ui/DKE-Strumbelj-Kononenko-Robnik.pdf>)
    4. Variable Importance (<https://arxiv.org/pdf/1801.01489.pdf>)
13. **Data pre-processing**: address missing values, remove useless instances, possibly discretize continuous features, and address other similar issues. This can be combined with feature selection to reduce the features to those relevant for the problem.
14. Why use LIME. SHAP over Correlation, ANOVA, GLM? - <https://christophm.github.io/interpretable-ml-book/the-future-of-interpretability.html>
15. Dashboard - Dash lets you make rich analytics web apps with only a few hundred lines of Python code. No JavaScript required. Dash is a user interface library for creating analytical web applications. Those who use Python for data analysis, data exploration, visualization, modelling, instrument control, and reporting will find immediate use for Dash.
16. Some applications of unsupervised machine learning techniques include:
    1. Clustering allows you to automatically split the dataset into groups according to similarity. Often, however, cluster analysis overestimates the similarity between groups and doesn’t treat data points as individuals. For this reason, cluster analysis is a poor choice for applications like customer segmentation and targeting.
    2. Anomaly detection can automatically discover unusual data points in your dataset. This is useful in pinpointing fraudulent transactions, discovering faulty pieces of hardware, or identifying an outlier caused by a human error during data entry.
    3. Association mining identifies sets of items that frequently occur together in your dataset. Retailers often use it for basket analysis, because it allows analysts to discover goods often purchased at the same time and develop more effective marketing and merchandising strategies.
    4. Latent variable models are commonly used for data preprocessing, such as reducing the number of features in a dataset (dimensionality reduction) or decomposing the dataset into multiple components.
17. *Wait times for preauthorized medical care have consequences for patients. 92 percent of the physicians surveyed said that the prior authorization process delays patient access to necessary care; and 78 percent reported that prior authorization can sometimes, often or always lead to patients abandoning a recommended course of treatment. Imagine the efficiencies that could be achieved if the manual workflow of pre-authorizations could be lifted and replaced with automatic approvals that rely on artificial intelligence solutions.*
18. Build a Logistic explainer
19. Try LIME on label/target encoder
20. This is how we do grid search hypertuning. [This](https://github.com/scikit-learn-contrib/categorical-encoding/blob/master/examples/grid_search_example.py) code is also good.
21. Save LIME explanation in html using *exp.save\_to\_file(‘xx.html’)*
22. How to deal with high cardinal feature procedure and diagnosis code
    1. AS IS – One Hot encoding
    2. [Target encoding](https://github.com/scikit-learn-contrib/categorical-encoding/blob/master/category_encoders/target_encoder.py)
       1. TargetEncoder (verbose=1, cols=None, drop\_invariant=False, return\_df=True, handle\_missing='error', handle\_unknown='error', min\_samples\_leaf=5, smoothing=5.0)
       2. means **=** df**.**groupby('x\_0')['y']**.**mean()
       3. df['x\_0'] **=** df['x\_0']**.**map(means)
       4. The trick is to “smooth” the average by including the average rating over all movies. In other words, if there aren’t many ratings we should rely on the global average rating, whereas if there enough ratings then we can safely rely on the local average. Refer [this](https://maxhalford.github.io/blog/target-encoding-done-the-right-way/).[this](https://www.wikiwand.com/en/Additive_smoothing) and [this](https://www.wikiwand.com/en/Bayes_estimator#/Practical_example_of_Bayes_estimators)



* 1. Kaggle winner’s comment on target encoding - *One of my main contributions to the team was Bayesian target encoding. The idea is to use bayesian statistics to encode the categorical variables. The cool thing is that we can encode not only the target mean, but other statistics like the median, mode, variance, skewness, and kurtosis using the same framework. We found that this style of target encoding outperforms the built-in LightGBM categorical encoding.* [*https://www.kaggle.com/mmotoki/avito-target-encoding*](https://www.kaggle.com/mmotoki/avito-target-encoding)*. Also,* [*this is an excellent link*](https://mattmotoki.github.io/beta-target-encoding.html) *to understand beta (bayesian) target encoding.*
  2. [Weight of Evidence encoding](https://github.com/scikit-learn-contrib/categorical-encoding/blob/master/category_encoders/woe.py)

1. Calculate WoE and IV metric for each category
   1. Excluding Age, Service quantity, Procedure and diagnosis code
   2. Include binned age and service quantity
   3. Include 95% ‘othered’ procedure and diagnosis code
   4. Include 100%ile procedure and diagnosis code
2. **Catboost** classifier
   1. Gradient boosting on decision trees
   2. [documentation](https://tech.yandex.com/catboost/doc/dg/concepts/python-reference_catboostclassifier-docpage/#python-reference_catboostclassifier)
   3. After setting a benchmark, it is time to explore your data
      1. Feature importance
      2. Feature interaction
      3. Per object feature importance (SHAP)
      4. Influential documents
      5. New feature evaluation
   4. Overfitting detector
   5. Missing values support
   6. <https://github.com/catboost/tutorials>
   7. https://github.com/catboost/catboost
3. AdaBoost with Decision Tree - <https://youtu.be/LsK-xG1cLYA>
4. In linear regression interpretation, we are assuming monotonicity (i.e. if one variable goes up/down, the output goes only in a particular direction)
5. Explanation techniques
   1. Decision tree surrogate models - <https://youtu.be/Q8rTrmqUQsU?t=1518>
   2. LIME
   3. SHAP – a silver bullet
      1. <https://medium.com/@gabrieltseng/interpreting-complex-models-with-shap-values-1c187db6ec83>.
6. <http://www.nada.kth.se/~ann/exjobb/sara_engardt.pdf>



1. Sensitivity analysis of your model – a important step
2. Meta frame
   1. Name
   2. Suitable name
   3. Description
   4. Values
   5. Cleaning and validation
   6. Encoding
   7. Transformation
   8. Data type
   9. Missing value
   10. Outlier
   11. Hypotheses (how it impacting the decision)
3. Bayesian Optimization
   1. **Bayes\_opt.** [Github](https://github.com/fmfn/BayesianOptimization). Example - [Simple Bayesian Optimization for LightGBM](https://www.kaggle.com/sz8416/simple-bayesian-optimization-for-lightgbm)
   2. Manual implementation function - <https://thuijskens.github.io/2016/12/29/bayesian-optimisation/>
   3. **Hyperopt** package. <https://anaconda.org/conda-forge/hyperopt>. Example [here](https://towardsdatascience.com/automated-machine-learning-hyperparameter-tuning-in-python-dfda59b72f8a). Another example [here](https://towardsdatascience.com/an-introductory-example-of-bayesian-optimization-in-python-with-hyperopt-aae40fff4ff0).
4. Simple logistic regression log odds plot

xs = np.linspace(-4,4,100)

pl.xlabel("Log odds of winning")

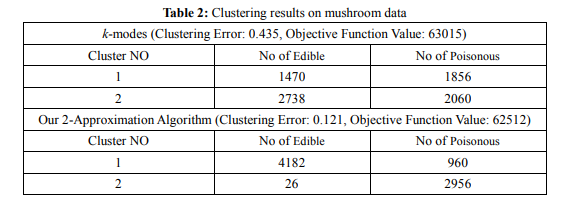
pl.ylabel("Probability of winning")

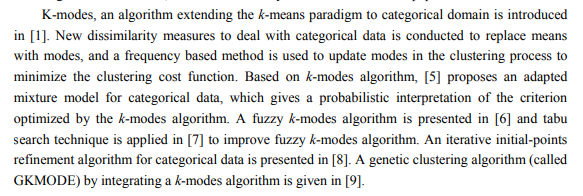
pl.title("How changes in log odds convert to probability of winning")

pl.plot(xs, 1/(1+np.exp(-xs)))

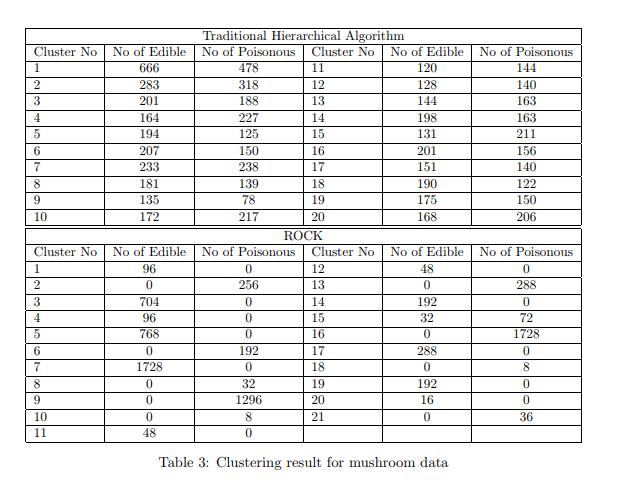
pl.show()

1. Jupyter markdown skills - <https://codeburst.io/jupyter-notebook-tricks-for-data-science-that-enhance-your-efficiency-95f98d3adee4>.





Excellent paper on clustering - <http://www.nada.kth.se/~ann/exjobb/sara_engardt.pdf>



* For Each models in Logistic, RandomForest, Xgboost, LightGBM, SVC and Catboost
  + Pre-process (SMOTE, Approved-NotApproved)
  + Feature Creation (Crosstab and factorplot analysis)
  + Encoding (Sparsed, Label, Target, WoE, Target+onehot, WoE+onehot)
  + Feature Selection (SelectKBest, SelectFromModel, IV, RFE, Feature\_Imp\_, ELI5, Corrmap)
  + Hyperparameter Tuning
  + Threshold Tuning
  + Advanced Ensembling (voting classifier)
  + Explanation (LIME, SHAP, ELI5)
* Logistic Hyperparameter Grid
  + param\_grid = [
  + {'classifier' : [LogisticRegression()],
  + 'classifier\_\_penalty' : ['l1', 'l2'],
  + 'classifier\_\_C' : np.logspace(-4, 4, 20),
  + 'classifier\_\_solver' : ['liblinear']}]
* Add Explanatory markdowns in the notebooks
* Clustering on Numerical encoded (target/woe) data - Minibatch Kmeans with Kmeans++, Heirarchical, DBSCAN,

<https://www.kaggle.com/esthergloriadawes/customer-segmentation>